ED 222 172 IR 010 405

AUTHOR	Renckly, Thomas R.; Orwig, Gary
TITLE	Curriculum Viewed as a Binary System: An Approach to the Determination of Sequence. A Project Report.
PUB DATE	81
NOTE	68p.; Paper presented at the Interservice/Industry Conference on Training and Equipment (3rd, Orlando, FL, November 29-December 2, 1981).
EDRS PRICE	MF01/PC03 Plus Postage.
DESCRIPTORS	*Computer Oriented Programs; Computer Programs; *Curriculum Design; *Educational Objectives; Flowcharts; Instructional Design; *Mathematical Models; Pacing; Pilot Projects; *Vertical Organization

ABSTRACT

A description of the development and application of a hierarchical/binary model by which a curriculum may be analyzed to determine alternative instructional sequences given particular instructional objectives and limiting constraints forms the body of this report. The background of the project as part of an effort by the U.S. Navy Recruiting Command to develop training programs for 16 closely related jobs is described; the complexity of the curriculum design process is discussed; the development of an instructional objectives hierarchy is outlined; and the characteristics and uses of binary matrices, transitive relations analysis, and directed graphs (digraphs) in the ranking of instructional objectives are detailed. The method employed to establish an instructional objectives hierarchy during the project is then recounted in step-by-step fashion, drawing on the preceding examination of ranking techniques. A computer algorithm which replicates the design process presented in the report is briefly discussed. Accompanying the text are 13 figures, 4 analytical tables, a 21-item bibliography, and 2 appendices--the first, a detailed computer flowchart for developing a sequence digraph from a set of curriculum objectives and the second, an Applesoft BASIC program listing based upon the flowchart presented in the first appendix. (JL)

ED222172

CURRICULUM VIEWED AS A BINARY SYSTEM: AN APPROACH TO THE DETERMINATION OF SEQUENCE

A PROJECT REPORT

by

Thomas R. Renckly
and
Gary Orwig

IR010405

CURRICULUM VIEWED AS A BINARY SYSTEM: AN APPROACH TO THE DETERMINATION OF SEQUENCE - A PROJECT REPORT

Thomas R. Renckly
Education Specialist
Curriculum Design Coordinator
U. S. Navy Recruiting Command/
Orlando, Florida

Gary Orwig, Ph.D.
Asst. Professor
College of Education
University of Central Florida
Orlando, Florida

ABSTRACT

Determining alternative curriculum sequences is a tedious task involving many individuals and analysis of large amounts of curriculum-related information. Because these tasks are not readily reducible to mathematical operations, and because educators and curriculum designers are generally not so inclined, computer intervention into this design process has been meager. The project reported herein describes the development and application of a model by which a curriculum may be analyzed to determine alternative instructional sequences based upon curriculum objectives and limiting constraints. The project's primary goal is to ultimately apply the model to the analysis and design of instructional sequences for 16 closely related courses currently under development by the U. S. Navy Recruiting Command.

Table of Contents

THE PROJECT

The U. S. Navy Recruiting Command has undertaken the task of developing training programs for 16 individual, yet closely related jobs (or billets) within the duty called recruiting. Although development of training programs is not new to the military, the approach to this particular curriculum design problem is. Due to the close interrelationship between and among each of the 16 courses under development, there is heavy reliance upon instructional sequence. For example, several competencies have been identified which overlap in many of the 16 courses. Because such overlaps parallel real-life recruiting practices, they were not avoided. It is educationally sound practice to structure student learning experiences so as to simulate reality as much as possible. Some educational psychologists believe this produces maximum learning transfer.

However, the payback comes in the form of an increased demand on curriculum sequencing. A non-sequitor or ill-sequenced curriculum can damage the realism of learning experience. It can also reduce the student's ability to internalize the concepts presented. The student may appear to peform satisfactorily in the school environment but become disoriented in trying to perform a similar task in the real-world environment. Thus the need for well-sequenced instruction.

The *classical* approach to determining acceptable instructional sequences has characteristically been human intuition. Such an approach is time-consuming in that it seldom produces an adequate sequence on the first attempt. Additionally, considerable work is involved with each iteration. In this current project, intuition is simply not sufficient for the task of aligning 16 courses into a unified sequence.

In any curriculum design problem, there are a myriad of variables which may dramatically affect the ultimate instructional sequence. However, a model exists in the literature which is capable of dealing with complex systems of interacting variables.[1] This model, known as Interpretive Structural Modeling (ISM), has been successfully applied to the sequencing of process elements in a number of design projects in the fields of engineering, agriculture and a host of other complex scientific and social problems.[2] Unfortunately, ISM has seen limited use in the field of education as a tool for planning and design. In fact, this writer has found only one such use. This has been accomplished primarily by Sato and his colleagues in Japan.[3,4] It is the intent of this project to adapt ISM to the instructional sequencing problem and build upon the work that has already been done in this area with the hope that successful development here may spawn more educational uses of ISM in this country.

As the initial project report, this paper will present some basic theory underlying the ISM concept as well as a method which shows great promise in assisting the curriculum designer in determining appropriate alternative instructional sequences.

COMPLEXITY IN THE DESIGN PROCESS

The instructional systems approach, or any systematic approach to instructional design for that matter, is anchored in mathematical modeling. It has long been recognized that a systems approach to instructional development is patterned after the scientific method[5] which is in itself a modeling approach.[6]

The question then arises as to why the design of instruction is not treated by a mathematical approach to approximating the shape and scope of a curriculum! In their text, Programmed Learning in Perspective, the authors allude to the mathematical character of curriculum. They describe a quasi-mathematical technique (termed the matrix technique) which is useful in determining optimum unit sequencing within programmed instructional material.[7] Davies further generalized this procedure, demonstrating its utility in optimizing presentation sequences for objectives of an entire course of instruction.[8] The logical extension of this work leads one to believe there may be a method by which a complex curriculum composed of disjointed competencies might be alternatively sequenced.

Successful instructional design models call for some sort of determination of sequence at some time during the design process. Often this is achieved through construction of objective trees (or hierarchies). In fact, instruction in the building of such hierarchies is often in great detail [9]-- testimony to its importance in the instructional design process. To anyone familiar with such a task, it is immediately obvious that instructional hierarchies are complex structures not only to build, but also to interpret. The casual observer is often unable to visualize the many possible sequencing strategies from the maze of lines displayed. Such insight requires a knowledge of the course content and at least some grounding in basic learning psychology. Yet, even if this prior knowledge is assumed, the task of choosing an appropriate sequence from all the possible sequences displayed on the hierarchy is still not easy. Mathematical modeling and operations research provide some interesting algorithms, however, which demonstrate the potential to assist in solving complex instructional sequencing problems.

In their paper Unified Program Planning, Hill and Warfield describe a method of reducing complex systems of elements (in our case, objectives) into a matrix which describes their mutual relationships.[10] They call this a *self-interaction matrix* because it contains information relating to the interaction of each element with itself and the others in the system. The authors define such a matrix as containing enough information to construct an objectives tree.

For this project, their matrix method is used in developing an objectives hierarchy from an initial set of course objectives. The worth of this matrix method is in its ability to produce a hierarchy which actually contains more information than hierarchies developed by other means. As an example of the kinds of information stored, and generally gleaned from typical objectives trees, consider the hierarchy of a hypothetical curriculum containing 15 interrelated objectives as shown in Figure 1 below.

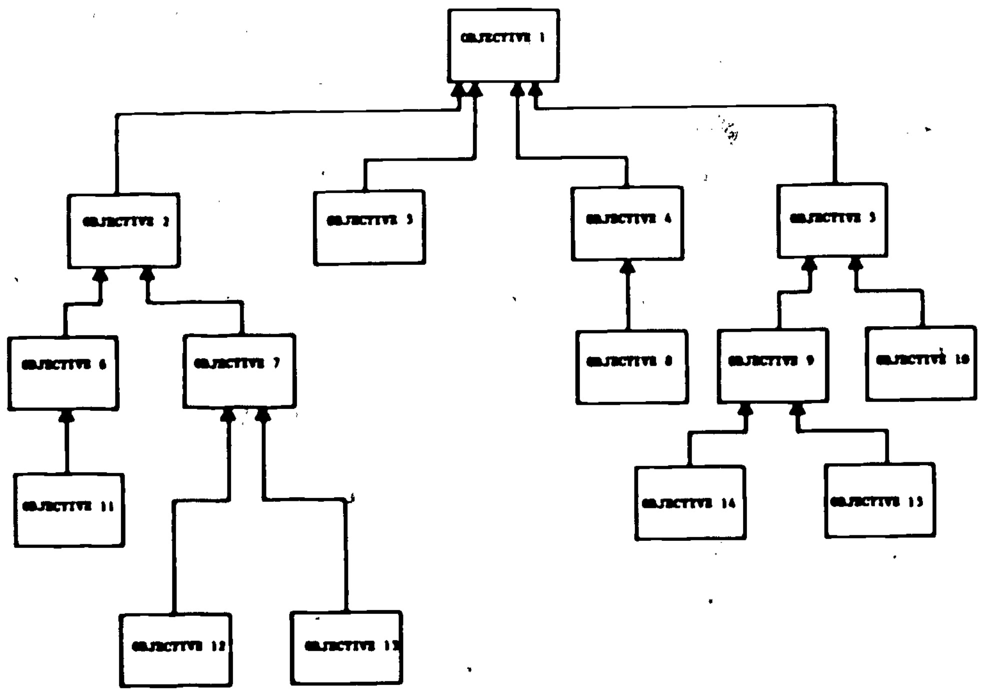


Figure 1. An objectives hierarchy containing 15 interrelated objectives for a hypothetical curriculum.

Several *bits* of information are implicitly stored in this hierarchy. For example, OBJECTIVE 1 appears to be the terminal objective for the curriculum. That is, all other objectives either directly or indirectly terminate at OBJECTIVE 1. Also, OBJECTIVEs 3, 8, 10, 11, 12, 13, 14 and 15 are at base levels with no supporting objectives. Thus, these are ideal starting points for sections or modules of instruction. Yet another *bit* of information available from the hierarchy is implied by the arrows connecting the various objectives. Their pattern indicates the existence of partitions between objective clusters (though such partitions are purely arbitrary). For example, one such partition could be OBJECTIVEs 11, 6, 2 and 1; another, OBJECTIVEs 13, 12, 7, 2 and 1; another, OBJECTIVEs 8, 4, and 1; still another, OBJECTIVEs 15, 14, 9, 5, and 1; etc. Although such partitions are arbitrary, these groupings give some indication of the amount of information potentially stored in an objective hierarchy. All these *bits* of information taken together represent a detailed picture of how each objective interacts with all the rest in this particular hypothetical curriculum.

Yet, a completely different class of interactions exists which also come to bear on a curriculum. This class contains such instruction-related items as resource constraints (money, manpower, and time), student needs, types of learning activities available to students to meet course objectives, types and timing of measurement tests, etc. Each of these has an effect on whether or not a given instructional sequence will work effectively. However, these interactions cannot be stored or displayed on a typical objectives hierarchy, such as that in Figure 1. Even by looking at the hierarchy, it is impossible to discern if such interactions were taken into consideration in the hierarchy's development.

Of course, this information could be superimposed onto the hierarchy; however, this could very easily complicate the diagram to the point that interpretation becomes impossible. The reason for this is that there seems to be an upper limit on the amount of information that one human can process and operate on at any given time. "Research tentatively shows that the amount of information man is capable of processing is limited, and more data...do not necessarily increase the quality of decisions in the same proportion."[12] It must be made clear at this juncture that the self-interaction matrix is not intended to replace the objectives tree, but only to enhance it. "The self-interaction matrix...is not as clear as the objectives tree for viewing the relationships among objectives, but it incorporates significant advantages in relating objectives to constraints, alterables and needs" inherent in the instructional system.[13] Thus, in this project, both the matrix and the objectives tree are utilized to their maximum advantages.

In actuality, the curriculum designer, the teaching staff, and the management personnel each recognize a different set of such interactions as mentioned above which impact on the curriculum. Thus, the designer must spend considerable time with teachers to develop an instructional hierarchy which takes into account as many of the ancillary interactions as possible. And when they finally come to an agreement on a reasonable teaching sequence, they may find (to their dismay) that the administration rejects the plan because of some constraining factor neither the designer nor the teachers knew about. Such situations are common and illustrate the need for a model which can contain and process much more curriculum-relevant information than is currently possible. The major requisites of such a model would have to be: convenience, simplicity and utility.

Convenience can be described as the ease of applying the model to the design problem. Simplicity refers to the quantity of information that must be provided by the user for the model's operation. And utility can be expressed as the model's adaptability to a general class of curriculum design problems - from the relatively simple task of sequencing information within a programmed text to the highly complex task of determining the sequence for effective learning in a *spiraled* "K through 12" educational network. ISM, the model used in this project, possesses these primary requisites in varying degrees and is thus a likely candidate for the curriculum design problem.

CHARACTERISTICS OF A BINARY MATRIX

Before detailing the results and current status of the project, we should first clarify the terms used. The literature on the subject is primarily mathematical. For this discussion, the mathematics have been simplified in some places, and eliminated altogether in others. In its place, intuitive arguments have been used. Readers interested in the actual mathematical derivations are referred to the work of Warfield.[14]

A <u>binary matrix</u> is a square array of elements whose values are either 1 or 0. If all the <u>main diagonal</u> elements (from upper left to lower right in the array) are 1s, the matrix is said to be <u>reflexive</u>. Thus, an <u>irreflexive</u> matrix has some 0s on its main diagonal. An irreflexive matrix must be made reflexive in order to be analyzed by the ISM matrix method. Fortunately, this is easily accomplished by adding to the irreflexive matrix an <u>identity matrix</u>. This is also a binary matrix with 1s along the main diagonal and 0s everywhere else.

The rows of a matrix are usually referred to by the letter i, while the columns are usually referred to by the letter j. Every matrix element occupies a position which is at the intersection of a row and column. Thus, any arbitrary element of a matrix can be referred to as the (i,j) element.

If a matrix element (i,j) and its "mirror-image" element (j,i) are the same value (either 1 or 0), then the matrix is said to be symmetric. The degree of symmetry depends upon how many elements (i,j) are matched to their "mirror-images". To illustrate this more clearly, note the mirror-image quality in the binary matrix in Figure 2 on both sides of the main diagonal. For clarity, the zeros have been removed.

$$
\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 1 &   &   & 1 & 1 \\
2 &   & 1 & 1 &   & 1 \\
3 &   & 1 & 1 &   &   \\
4 & 1 &   &   & 1 & 1 \\
5 & 1 & 1 &   & 1 & 1 \\
\hline
\end{array}
$$

Figure 2. Mirror Image Symmetry Above and Below the Main Diagonal (dashed)

A binary matrix may have a few assymetric points and still be considered symmetric for purposes of this method if the number of assymetric points are small. In reality, an assymetric matrix yields the best instructional hierarchy. Thus, the degree of assymetry in the matrix determines the richness of the resulting hierarchy. However, this depends upon the nature of the objectives under consideration and the nature of the interactions among objectives - both of which are dependent on the type of curriculum being designed.

TRANSITIVE RELATIONS AND DIRECTED GRAPHS

In determining an appropriate curriculum sequence, considerable thought must be given to how each instructional objective relates to all other objectives in the curriculum. During the so-called "front-end analysis" phase of a design project, relationships between what the student needs and what the curriculum will offer to meet those needs are more likely to be philisophical intuitions than rigorous proofs. The mathematical character of ISM, however, requires a more detailed analysis of such relationships. These relationships are logical rather than mathematical.

Consider the logical relationship among three objectives (_a_, _b_, and _c_) as illustrated in Figure 3. Figure 3A shows that objective _a_ relates to objective _b_, and that _b_ relates to _c_. However, objectives _a_ and _c_ are not directly related to one another. Clearly, if objective _b_ were removed from the curriculum, objectives _a_ and _c_ would exist as isolated entities. Such a relation among objectives is called _intransitive_ because there is no direct relation or, or connection, between objectives _a_ and _c_.

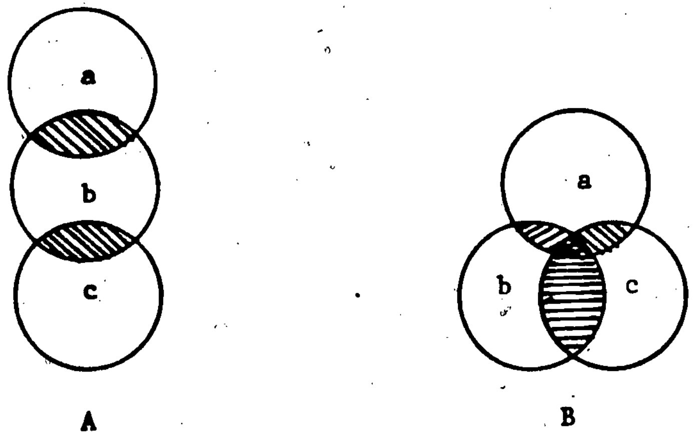


Figure 3. Two Types of Relationships Among Objectives a, b, and c.

Figure 3B, on the other hand, indicates that all objectives are directly related to each other. If any one of them is removed, the remaining two are still linked together through a binding relationship. A transitive relation is one in which each objective relates, or is somehow linked to the others in the group.

Though we have used the term "relation" numerous times, we have not yet clearly defined it. A relation is a phrase or term that shows how two or more elements (or objectives) interconnect, or link, to one another. Whether or not a relation is transitive depends not so much on what relation is used, as on the situation in which it is used.

For example, consider the relation "is contained within". If a "is contained within" b, and if b "is contained within" c, then it follows that a "is contained within" c. We can visualize this relation in Figure 4. Any objectives a, b, and c for which this relation holds true is considered a transitive set of objectives. It must be borne in mind, however, that even though the *relation* is transitive, not all *objectives* will suit it. If one particular relation is not transitive across an entire set of objectives under consideration, a relation that does apply must be found. Each new relation chosen, of course, must be similarly tested to insure transitivity within the entire objective set.

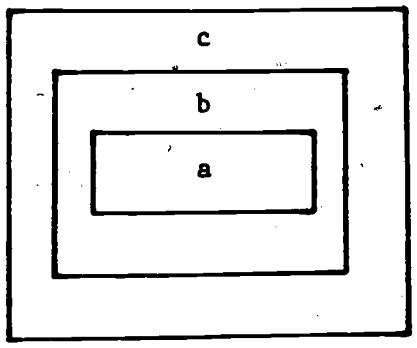


Figure 4. Visualization of the relation "is contained within".

Some relations are intransitive in all but the most specific of situations. For example, Warfield (1981), has reported that the relation "obeys" fails the transitivity test:[15] if a "obeys" b, and if b "obeys" c, a may not necessarily "obey" c. In fact, many relations are situation specific. They must be carefully considered in the context of the entire objective set.

Once a transitive relation has been identified, it remains to be discovered how the relation specifically affects each pair of objectives. Does, for example, the relation link objective a to objective b, or vice versa? A simple example should serve to illustrate this point. Consider the transitive relation *"depends upon prior accomplishment of"*. If objective a *"depends upon prior accomplishment of"* objective b, then clearly, b cannot possibly *"depend upon prior accomplishment of"* objective a. In addition to illustrating assymetry, this example also illustrates the concept of directability. In the above example, an arrow could be drawn between objectives a and b with the arrowhead pointing toward objective a to show that a *"depends upon prior accomplishment of"* b.

If all such directed relations between objectives are considered, a picture of the interactions can be obtained. Such a picture is known as a directed graph. Warfield has shown that any directed graph or *digraph* possesses an associated binary matrix.[16] A given binary matrix, however, may produce a number of alternative digraphs. Any one of them could be used as an objectives hierarchy to describe the interrelationships among instructional objectives. The binary matrix needed to produce the digraph is called the reachability matrix.

If transitive and assymetric, this matrix can be manipulated to produce a digraph (otherwise known as an objectives hierarchy). The procedure, described by Warfield, requires the formation of tables consisting of various arrangements of objectives.[17] The actual procedure followed for this project will be described in greater detail in the next section.

THE PROJECT'S METHOD

The process of generating a digraph from a set of curriculum objectives is a straight-forward approach composed of the following steps:

1. Identify the objectives of the curriculum.
2. Determine a transitive relation which applies to the objectives in the context of the instructional situation.
3. Place objective relations into a matrix format - termed a <u>self-interaction matrix</u>.
4. Manipulate the matrix into a suitable form - termed a <u>reachability matrix</u>.
5. Re-order the rows and columns of the reachability matrix and partition it to reflect hierarchial levels - termed a <u>modified reachability matrix</u>.
6. Compute a hierarchy (or <u>digraph</u>) from the modified reachability matrix.

The curriculum design project described in this paper follows this six step process for generating hierarchies and determining instructional sequences. Since the approach is both complex and time consuming, computer algorithms have been designed to perform most of this work. The remainder of this paper details the process followed in the Navy curriculum project.

Step 1. After the front-end analysis had been completed for the 16 courses under development, a listing of tasks required for training were identified. And from these, a series of learning objectives were developed for each course. One course was used for the pilot study in this project.

Step 2. The transitive relation *is necessary to accomplish* was agreed upon by the subject matter specialists, the curriculum design staff and the approving board for curriculum development. This relation was used in the analysis of the relationships between every possible pair of objectives. Since 18 objectives were originally identified for training in the pilot course, 18 x 18 (=324) distinct objective pairs were analyzed via the agreed upon relation.

Step 3. For each of the 324 objective pairs, a 1 was placed into the corresponding cell of a matrix, if the relation was true. If, however, the relation was false for a particular pair, a 0 was placed in the appropriate matrix cell. The self-interaction matrix which resulted is shown in Figure 5.

Objective Number

Objective Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	1	0
2	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
5	0	0	1	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	1	1	0	0	1	0	0	1	1	0	1	1
8	0	0	0	0	1	0	1	1	0	1	1	0	0	1	0	0	0	0
9	0	0	1	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0
11	0	0	0	0	1	0	1	0	0	0	1	1	0	1	0	0	1	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
13	0	0	1	1	1	0	0	0	1	1	0	0	1	0	0	0	0	0
14	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
18	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Figure 5. A Self-Interaction Matrix for the Pilot Course

Warfield describes an algorithm with which a computer can be programmed to accomplish this data entry step with reduced effort on the part of the user.[18] The algorithm has been modified for use in this project. After creation of the self-interaction matrix of Figure 5, it was loaded into a BASIC language microporcessor via a prompting routine developed by Orwig. The flowchart of this routine is shown in Figure 6.

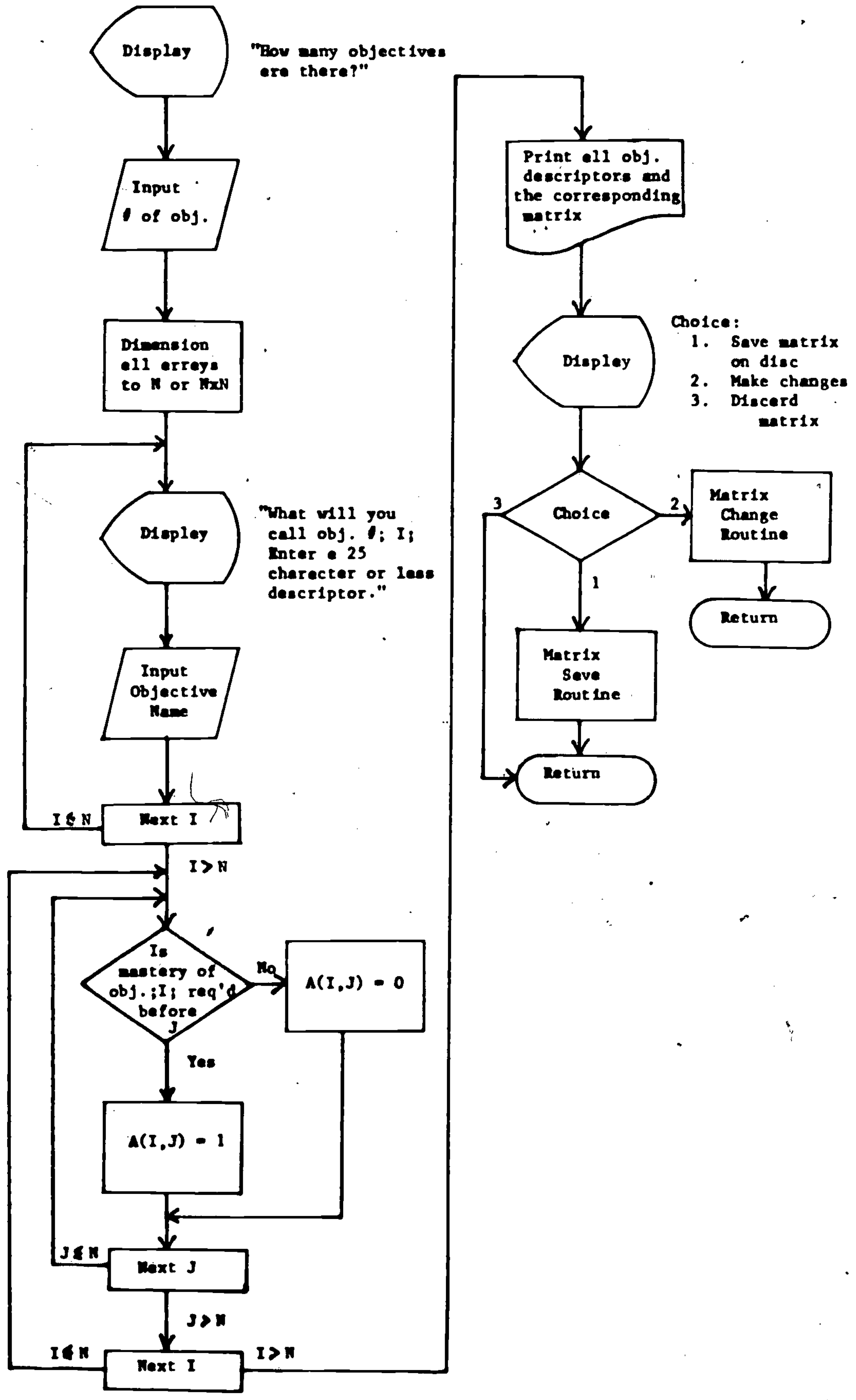


Figure 6. A Prompting Routine for Matrix Data Entry

Step 4. The self-interaction matrix of Figure 5 must be manipulated into a reachability matrix before further analysis can be performed. This manipulation involves raising the self-interaction matrix (A) to successive powers (squared, cubed, etc, by Boolean multiplication) until the following equality is met: $A^n = A^{n+1}$. An algorithm used to accomplish this multiplication process is presented in flowchart form in Figure 7. (A flowchart of the entire computer program developed by the authors appears in the Appendix). According to theory, if there are N objectives in the matrix, the reachability matrix will be derived in N-1 or less iterations.[19] The self-interaction matrix for the pilot course (Figure 8) was converted to reachability form in four iterations. In other words, the matrix of Figure 8 multiplies out in four iterations to form the reachability matrix in Figure 9, which satisfies the equality: $A^3 = A^4$.

Objective Number

Objective Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	1	0
2	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
5	0	0	1	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	1	1	0	0	1	0	0	1	1	0	1	1
8	0	0	0	0	1	0	1	1	0	1	1	0	0	1	0	0	0	0
9	0	0	1	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
11	0	0	0	0	1	0	1	0	0	0	1	1	0	1	0	0	1	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
13	0	0	1	1	1	0	0	0	1	1	0	0	1	0	0	0	0	0
14	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
18	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Figure 8. The Self-Interaction Matrix of Figure 5.

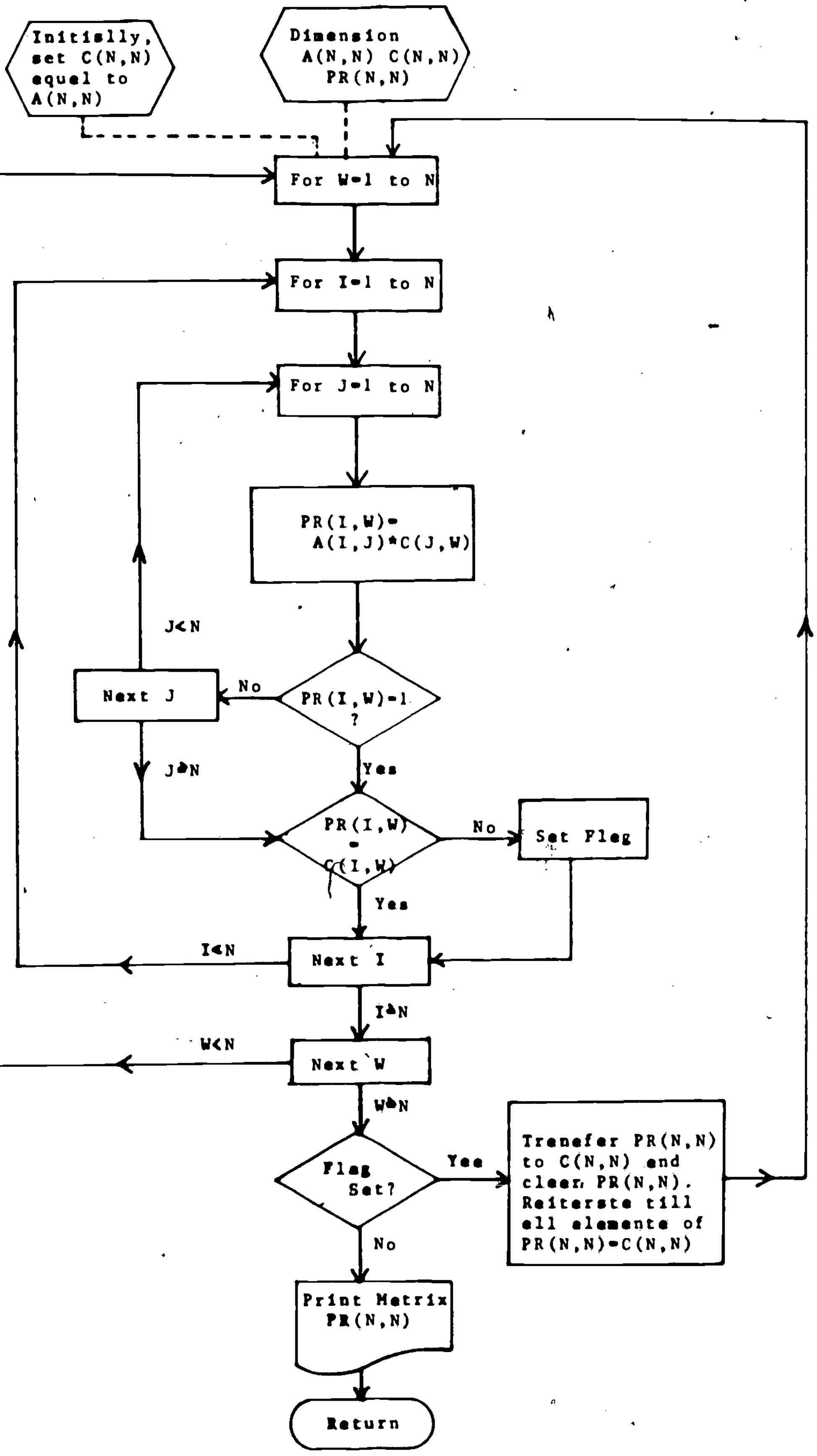


Figure 7. An algorithm to convert a self-interaction matrix A(N,N) into a reachability matrix PR(N,N)

Step 5. The purpose of this step is to partition the reachability matrix into submatrices which reflect the levels within the instructional hierarchy. The resulting partitioned matrix will be the reachability matrix modified by row and column interchanges. To determine the eventual order of this interchange, a table is created which contains a reachability set, an antecedent set and the product (or intersection) of both sets.

Objective Number

Objective Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
2	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
5	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
8	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
9	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
10	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
11	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
13	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
14	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
17	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1
18	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1

Figure 9. The Reachability Matrix Derived From the Matrix of Figure 8.

The reachability set for element 1 is found by inspecting row 1 of the reachability matrix (Figure 9). Every 1 in row 1 corresponds to a column index, and every such column index will be in the reachability set of element 1. To find the antecedent set of element 1, inspect column 1. To every entry of 1 in column 1, there is a corresponding row index; and the set of such row indices is the antecedent set of column 1.

Each row and colum is similarly considered in turn thus producing a table of reachability and antecedent sets for each row of the matrix.

In Figure 9, the row and column indices (1-18) are used to identify the respective elements of the reachability and antecedent sets. Table 1 is constructed from Figure 9 by inspection.

From Table 1, it is immediately apparent that the only rows for which the set product equals the reachability set are rows 6 and 15. These two rows are therefore removed from the table along with all references to numbers 6 and 15 everywhere else in the table. Thus, rows 6 and 15 from the reachability matrix (Figure 9) become the first two rows of the modified reachability matrix. These two rows will be considered the top level in the instructional hierarchy (or digraph).

Ordinarily, the references to rows 6 and 15 can simply be erased from the table, and the next iteration begun. For the purpose of illustration here, however, each new (reduced) table will be enumerated.

Removal of all 6s and 15s results in the reduced form of Table 2. This time, the reachability set R(s) and set product columns match for rows 4 and 12. As before, these rows are removed from the table and the reachability matrix to become the second level in the modified matrix. Again, removing all references to 4 and 12 from the above table results in the formation of Table 3.

From Table 3, the third level of the modified matrix is shown to be composed of rows 2, 16, 3, 5, 7, 8, 9, 10, 11, 13, 14, 17, and 18. Deleting all these references from Table 3 results in the formation of Table 4.

Table 1. A Reachability Table

ROW INDEX(S)	REACHABILITY SET R(S)	ANTECEDENT SET A(S)	SET PRODUCT R(S) ∩ A(S)
1	1 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18	1	1
2	2 4 15	2	2
3	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
4	4 15	1 2 3 4 5 7 8 9 10 11 13 14	4
5	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
6	6	1 6	6
7	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
8	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
9	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
10	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
11	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
12	12 15	1 3 5 7 8 9 10 11 12 13 14 16 17 18	12
13	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
14	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
15	15	1 2 3 4 5 7 8 9 10 11 12 13 14 15 16 17 18	15
16	12 15 16	16	16
17	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
18	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18

Table 2. Reduced Table - Level 1 Removed

ROW INDEX(S)	REACHABILITY SET R(S)	ANTECEDENT SET A(S)	SET PRODUCT R(S) ∩ A(S)
1	1 3 4 5 7 8 9 10 11 12 13 14 17 18	1	1
2	2 4	2	2
3	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
4	4	1 2 3 4 5 7 8 9 10 11 13 14	4
5	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
7	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
8	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
9	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
10	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
11	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
12	12	1 3 5 7 8 9 10 11 12 13 14 16 17 18	12
13	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
14	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
16	12 16	16	16
17	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
18	3 4 5 7 8 9 10 11 12 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18

Table 3. Reduced Table - Level 2 Removed

ROW INDEX(S)	REACHABILITY SET R(S)	ANTECEDENT SET A(S)	SET PRODUCT R(S) ∩ A(S)
1	1 3 5 7 8 9 10 11 13 14 17 18	1	1
2	2	2	2
3	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
5	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
7	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
8	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
9	3 4 5 7 8 9 10 11 12 13 14 15 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
10	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
11	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
13	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
14	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
16	16	16	16
17	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18
18	3 5 7 8 9 10 11 13 14 17 18	1 3 5 7 8 9 10 11 13 14 17 18	3 5 7 8 9 10 11 13 14 17 18

Note that only row 1 remains to make up the fourth and final level of the modified matrix. The resulting modified matrix is shown in Figure 10. The heavy black squares clarify various submatrices which denote the four levels identified from the tables. Note that both row and column designations in the modified matrix have been identically interchanged. This is automatically accomplished by the computer algorithm.

Table 4. Reduced Table - Level 3 Removed

ROW INDEX (S)	REACHABILITY SET R(S)	ANTECEDENT SET A(S)	SET PRODUCT R(S) ∩ A(S)
1	1	1	1

Objective Number

$$
\begin{array}{c|cccccccccccccccccc|l}
 & 15 & 6 & 4 & 12 & 2 & 16 & 3 & 5 & 7 & 8 & 9 & 10 & 11 & 13 & 14 & 17 & 18 & 1 & \\
\hline
15 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{—Level 1} \\
6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{—Level 2} \\
12 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
2 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
16 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
3 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
5 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
7 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
8 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
9 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \text{—Level 3} \\
10 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
11 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
13 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
14 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
17 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
18 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \text{—Level 4}
\end{array}
$$

Objective Number

Figure 10. The Modified Reachability Matrix Containing Four Hierarchial Levels

The dashed lines within the level 3 submatrix identify constituents, or interior links, within the level. The largest of the three constituents is called a universal submatrix because it contains all 1s indicating that each of the associated objectives in that submatrix are mutually reachable to and from each other. In the literature, this is more commonly known as a *maximal cycle*. The dashed lines to the left of each of the four heavy-lined submatrices outline what we in the project have termed communication submatrices which essentially describe how one level communicates with the level above it. These submatrices become useful in determining paths in the eventual digraph.

Step 6. At this step, all required information exists in the modified reachability matrix to compute the digraph. Warfield has noted that a given reachability matrix does not produce a unique digraph.[20] This implies that more than one digraph can be constructed from the reachability matrix of Figure 10. All the digraphs constructed in this project are generalized digraphs which are actually composites of all the possible digraphs contained in the reachability matrix.

The construction method is illustrated using Figure 10 for reference. The following illustration represents the process used to manually construct the digraph from the modified matrix. The computer algorithm accomplishes this entire process in a manner which is *transparent* to the user. The process is presented here for those who wish to develop their own algorithms.

Begin by laying out each of the four levels identified by the heavy-lined submatrices (levels) and starting at the bottom of the matrix. Level 4 contains only row 1. Level 3, the largest level, contains rows 2, 16, 3, 5, 7, 8, 9, 10, 11, 13, 14, 17, and 18. Level 2 contains rows 4 and 12. And Level 1, the highest level, contains rows 15 and 6. By referring to the dashed submatrices (communication submatrices) to the left of each level submatrix, connecting paths between the objectives of one level and the objectives of each higher level on the hierarchy can be determined.

For example, the level 4 communication submatrix (bottom row in Figure 10) has the following pattern: (0 0 1 1 1 1 1 1 1 1 1 1 1).

This pattern matches the patterns of rows 3, 5, 7, 8, 9, 10, 11, 13, 14, 17, and 18 in the third level. Thus, a connecting path from objective 1 to each of those mentioned above can be drawn on the digraph. Note here that no connecting path exists between objectives 1 and 16 or 1 and 3 because their communication patterns do not match.

On level 3, there are three separate parts (or *constituents*) within the level. One constituent is composed of objective 2; another is composed of objective 16; and the third is composed of objectives 3, 5, 7, 8, 9, 10, 11, 13, 14, 17, and 18. As stated earlier, this third constituent is called a maximal cycle. Thus, interconnection paths can be drawn on the digraph between objectives 3, 5, 7, 8, 9, 10, 11, 13, 14, 17, and 18.

To get from level 3 to level 2, the level 3 communication submatrix (the dashed matrix to the right of the level 3 submatrix) is analyzed against the level 2 submatrix. Since level 3, as was shown, possesses three separate and unique constituents, there are three unique communication patterns to consider. For instance, the maximal cycle constituent (the largest within level 3) has a communication pattern of (1 1). This pattern matches the 1s in both row 12 and row 4 of level 2. Thus, connection paths can be drawn on the digraph from any member of the level 3 constituent to each of objective 12 and 4 in level 2. It is suggested here that only one member from level 3 be connected to level 2 since each member of the maximal cycle is already connected to all others in that constituent (by virtue of it being a maximal cycle set). In addition, a single connecting path allows the resulting digraph to appear considerably more simple.

However, the choice to do, or not to do, this is completely arbitrary. Also in level 3, the row 16 communication pattern (0 1) matches only row 12 in level 2, while row 2's communication pattern matches only row 4. Thus, two more connecting paths can be drawn. Continuing in this manner, a complete digraph can be drawn to represent the reachability matrix. The finished digraph is shown in Figure 11.

We should digress here for a moment to make an important point. Tatsuoka contends that a digraph can be constructed merely by analyzing the self-interaction matrix (he terms it the *adjacency matrix*).[21] This writer, however, believes that although Tatsuoka's contention is valid and logically consistent, the adjacency matrix contains only enough information for one unique digraph, whereas, the reachability matrix yields a more generalized digraph. In a manner of speaking, the reachability matrix is a composite of a family of adjacency matrices. This is intuitively true since the reachability matrix is computed by raising the adjacency matrix to consecutive powers.

This can also be shown mathematically. Take, for example, the number 16. There are two numbers who consecutive products will equal 16 - they are, of course 2 (2x2x2x2) and 4 (4x4). Both consecutive products result in the same number - 16. However the original numbers 2 and 4 are obviously not the same. Assume for the moment that 2 and 4 are adjancency matrices. Each will yield a certain digraph with unique interconnecting paths. They may turn out to be identical. But, chances are they will each be slightly different - one containing perhaps more paths than another. However, 16 can be thought of as a reachability matrix since it is a multiple of 2 and 4, and will therefore produce a digraph containing all the paths produced from 2 as well as 4.

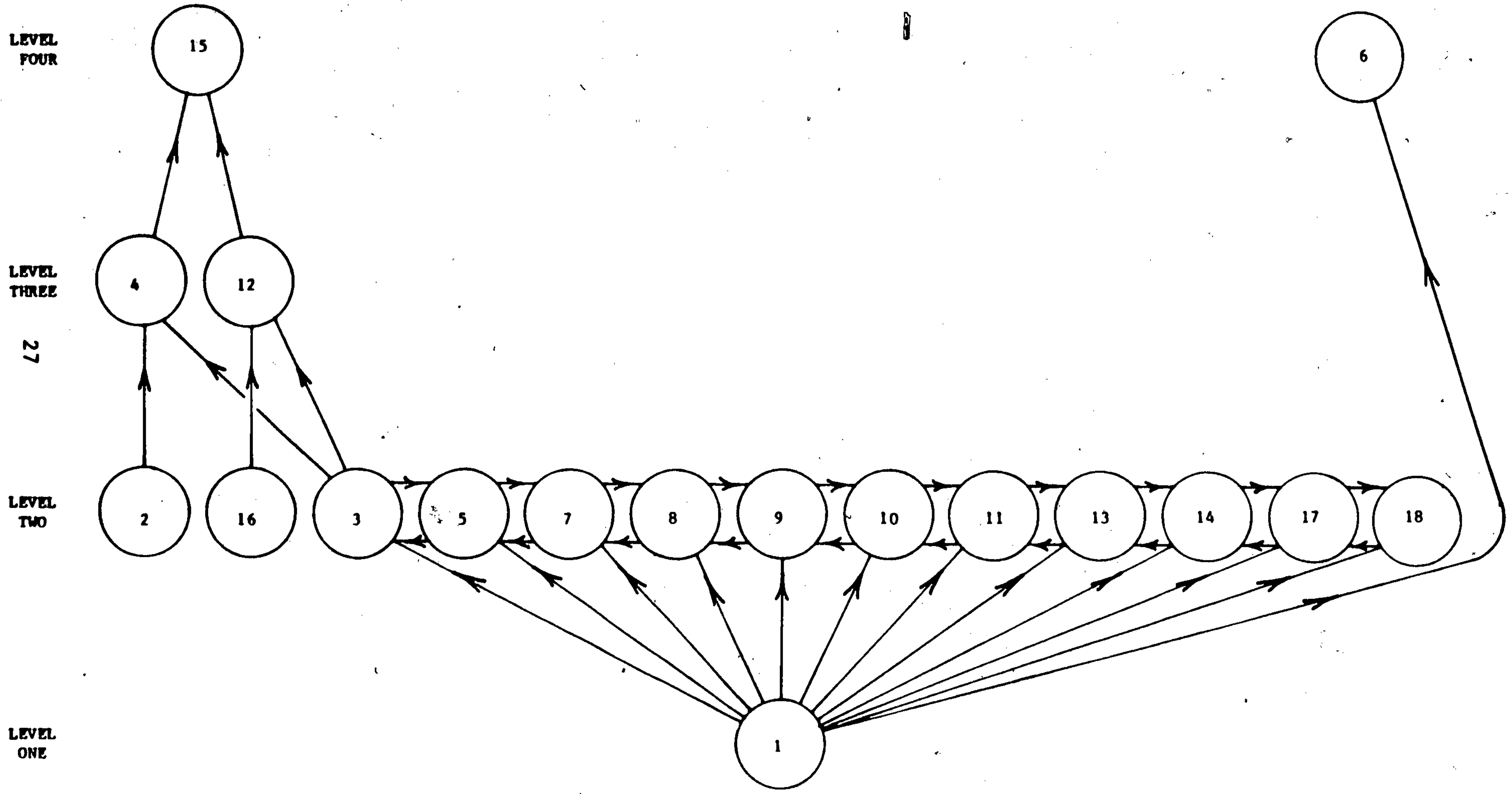


Figure 11. A Digraph for the Pilot Curriculum

Thus, the reachability matrix is said to produce a composite or *generalized* digraph.

A digraph computed from the adjacency matrix will undoubtedly be more simplified than one derived from the reachability matrix, although the level of complexity does not begin to become a hinderance until very large numbers of objectives (40 or more) are to be manipulated. In other words, the digraph derived from the reachability matrix will usually contain more paths than that computed from the adjacency matrix.

Each path on the digraph can be thought of as a legitimate transition from one objective to another within the curriculum. Looking at the digraph in this way, one can begin to see that by developing such a transition-laden digraph yields a more fertile data base from which alternative instructional sequences may be derived. With that, we'll return to the project description.

In the pilot project, it was recognized that the digraph of Figure 11 could be redrawn to yield more meaningful information to the curriculum designer. This alternate digraph is shown in Figure 12. Note in this figure that the maximal cycle constituent of level 3 is represented by a bi-directional circle interlocked via objective 1. From the viewpoint of the acutal course curriculum there is, in fact, a great deal of coherence among objectives 1, 3, 5, 7, 8, 9, 10, 11, 13, 14, 17 and 18. Thus, it is not coincidental that such a pattern has emerged. Note also that objective 6 can only be reached by objective 1. Therefore, any instruction concerning objective 6 must rely on information presented during instruction on objective 1 - if, that is, the students are to see a logical transition from one lesson to the next.

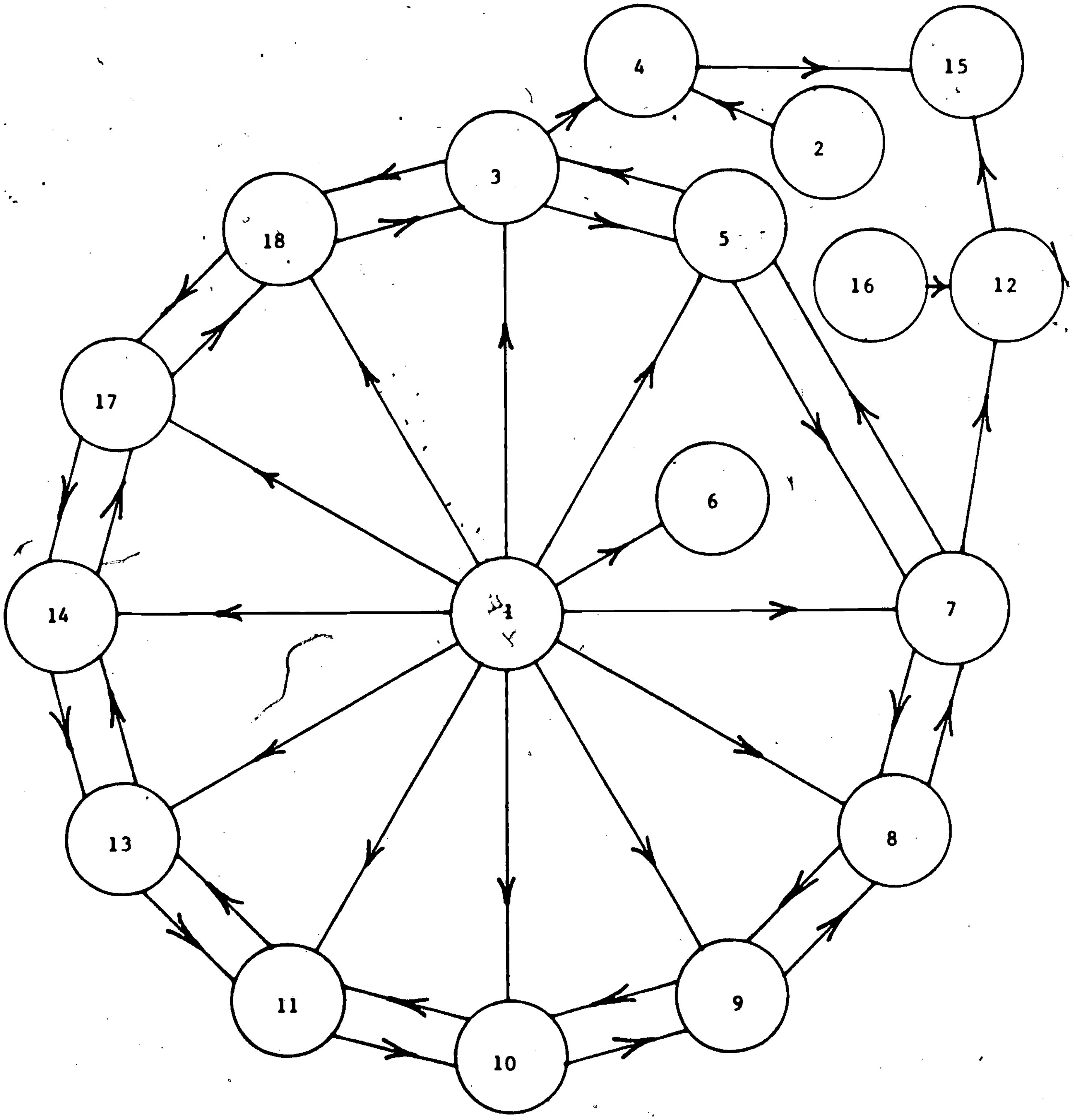


Figure 12. ALTERNATIVE DIGRAPH FOR PILOT CURRICULUM

Since objectives 1 and 6 appear isolated from the rest, instruction relating to these two objectives could very easily form a module of instruction. Indeed, other modules begin to emerge from the digraph upon closer inspection. It will be left to the reader to discover other such modules.

This, in and of itself, is a remarkable tool for the curriculum designer - to be able to identify "natural" groupings of objectives via mathematical analysis. However, this is merely a fringe benefit of the ISM procedure as we have designed it. As the computer analyzes the reachability matrix and its communication patterns, a data base is formed which contains all possible legitimate transitions from any given objective to any other. Once computed, this data base is used for comparison with a user's transition selections. A user can, in fact, experiment with various instructional sequences - transitioning from one objective to another until an entire course is created. By comparing user-selected transitions with the permissible transitions stored in memory, the computer will inform the user if a particular instructional sequence is, or is not, advisable. It will even printout the sequence created by the user in hard copy, if a printer is attached. Figure 13 is an actual, though partial, computer printout of the interactive instructional sequence creation routine.

WITH WHICH OBJECTIVE WOULD YOU LIKE TO START THE SEQUENCE?   1

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

3 5 7 8 9 10 11 13 14 17 18   ? 7

OK. $1 \rightarrow 7$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

3 5 8 9 10 11 13 14 17 18   ? 3

OK. $1 \rightarrow 7 \rightarrow 3$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

4 5 7 8 9 10 11 12 13 14 17 18   ? 4

OK. $1 \rightarrow 7 \rightarrow 3 \rightarrow 4$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

15   ? 2

THIS OBJECTIVE IS OUT OF SEQUENCE. DO YOU STILL WANT TO SELECT IT (Y OR N)? Y

OK. HOWEVER, IT WILL BE FLAGGED TO REMIND YOU IT'S OUT OF SEQUENCE.

$1 \rightarrow 7 \rightarrow 3 \rightarrow 4 \rightarrow \boxed{2}$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

4   ? 4

OK. $1 \rightarrow 7 \rightarrow 3 \rightarrow 4 \rightarrow \boxed{2} \rightarrow 4$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

15   ? 8

Figure 13. An Interactive Instructional Sequence Dialog Between User and Computer

THIS OBJECTIVE IS OUT OF SEQUENCE. DO YOU STILL WANT TO SELECT IT (Y OR N)? Y

OK. HOWEVER IT WILL BE FLAGGED TO REMIND YOU IT'S OUT OF SEQUENCE

$1 \to 7 \to 3 \to 4 \to \boxed{2} \to 4 \to \boxed{8}$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

3 5 7 9 10 11 13 14 17 18 ? 18

OK. $1 \to 7 \to 3 \to 4 \to \boxed{2} \to 4 \to \boxed{8} \to 18$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

3 5 7 8 9 10 11 13 14 17 ? 16

THIS OBJECTIVE IS OUT OF SEQUENCE. DO YOU STILL WANT TO SELECT IT (Y OR N)? Y

OK. HOWEVER IT WILL BE FLAGGED TO REMIND YOU IT'S OUT OF SEQUENCE.

$1 \to 7 \to 3 \to 4 \to \boxed{2} \to 4 \to \boxed{8} \to 18 \to \boxed{16}$

THE FOLLOWING TRANSITIONS ARE ADVISED. CHOOSE ONE OR ENTER ZERO TO END THE SEQUENCE:

12 ? 0

OK. HERE IS THE CURRICULUM SEQUENCE YOU HAVE CREATED:

$1 \to 7 \to 3 \to 4 \to \boxed{2} \to 4 \to \boxed{8} \to 18 \to \boxed{16}$

DO YOU WANT TO CREATE ANOTHER SEQUENCE (Y OR N)? N

OK. BYE FOR NOW.

Figure 13. An Interactive Instructional Sequence Dialog Between User and Computer (Continued)

LIMITATIONS WITHIN A CURRICULAR SYSTEM

Naturally, the ultimate decision as to how a curriculum is to be arranged rests with the managers or administrators of the curriculum. It has been this writer's experience that a major deficiency of front-end analysis is the inadequate attention paid to the interplay among the numerous internal and external constraints and limitations placed upon a given curriculum. Limitations such as facilities, personnel, time, money, social factors, etc., if not anticipated in advance of establishing a curriculum sequence, could result in the ultimate alteration of an otherwise logical instructional sequence.

It is admittedly a complex task to consider the effects of all possible limitations affecting a curriculum without some means to organize and manipulate very large amounts of data. The project described in this paper has illustrated a method with the power to expand and accomodate the analysis of such limitations - and thus produce an ultimate curriculum sequence which is sensitive to those limitations.

The ultimate goal of this project is to develop an integrated curriculum for 16 closely related courses. Each course possesses certain characteristic limitations which are either reinforced or overcome by the remaining courses. It is desired that this project will produce a curriculum which will reconcile the majority of those limitations. Such a goal is common to curriculum designs both in the military and civilian sectors of education. In that respect, at least, those of us associated with this project feel a bond with educators in every sector of society.

ABOUT THE AUTHORS

Mr. Thomas R. Renckly, Education Specialist, U. S. Navy Recruiting Command. Coordinator for Curriculum Design at the Navy Recruiting Orientation Unit, Educational Research and Systems Development Department, Orlando, Florida. Currently pursuing a doctorate in Curriculum and Instruction at the University of Florida.

Dr. Gary Orwig, Asst. Professor in Instructional Technology with the College of Education at the University of Central Florida. Teaches courses in computer applications in instructional technology; actively engaged in research concerning interactive video and instructional design.

BIBLIOGRAPHY

1. Hill, Douglas J. and Warfield, John N. "Unified Program Planning," IEEE Transactions on Systems, Man, and Cybernetics, SMC-2, No. 5, Nov. 1972, pp. 610-621

2. Delp, P., Thesen, A., Motiwalla, J., and Seshardi, N. System Tools for Project Planning, International Development Institute, University of Indiana, Bloomington, Indiana, 1977, pp. 92-103

3. Sato, Takahiro. "Hierarchical Display of Networks of Teaching Elements Using the Interpretive Structural Modeling Method" IECE Transactions on Educational Technology, 1978-04, pp. 27-30 (manuscript in Japanese)

4. Sato, Takahiro. "Determination of Hierarchical Networks of Instructional Units Using the Interpretive Structural Modeling Method," Educational Technology Research, No. 3, 1979, pp. 67-75 (In English)

5. Bobbitt, Franklin. How To Make A Curriculum, Boston: Houghton Publishing Co., 1924

6. Nadler, Gerald. Work Design: A Systems Concept, Homewood, Illinois: Richard D. Irwin Inc., (Revised) 1970, pp. 21-47

7. Thomas, C. A., Davies, I.K., Openshaw, D., and Bird, J. Programmed Learning in Perspective: A Guide To Program Writing, Chicago: Educational Methods Inc., 1963

8. Davies, Ivor K. Competency Based Learning: Technology, Management and Design, New York: McGraw-Hill, 1973, pp. 92-102

9. Esseff, Peter J. and Mary S. Educational Systems For The Future (Mediated Presentation), 1978, (4th printing)

10. Op. Cit., Hill and Warfield, p 613

11. Op. Cit., Hill and Warfield, p 613

12. Op. Cit., Nadler, p 524

13. Op. Cit., Hill and Warfield, p 614

14. Warfield, John N. Structuring Complex Systems, Columbus: Battelle Institute, Monograph No. 4, April, 1974

15. Warfield, John N. personal communication, February, 1981

16. Op. Cit., Warfield, Chapter 2, p 9

17. Warfield, John N. Societal Systems: Planning, Policy and Complexity, New York: John Wiley & Sons, 1976, pp. 276-284

18. Op. Cit., Warfield, Structuring Complex Systems, Chapter 4, pp. 1-8; Chapter 5, pp. 1-9

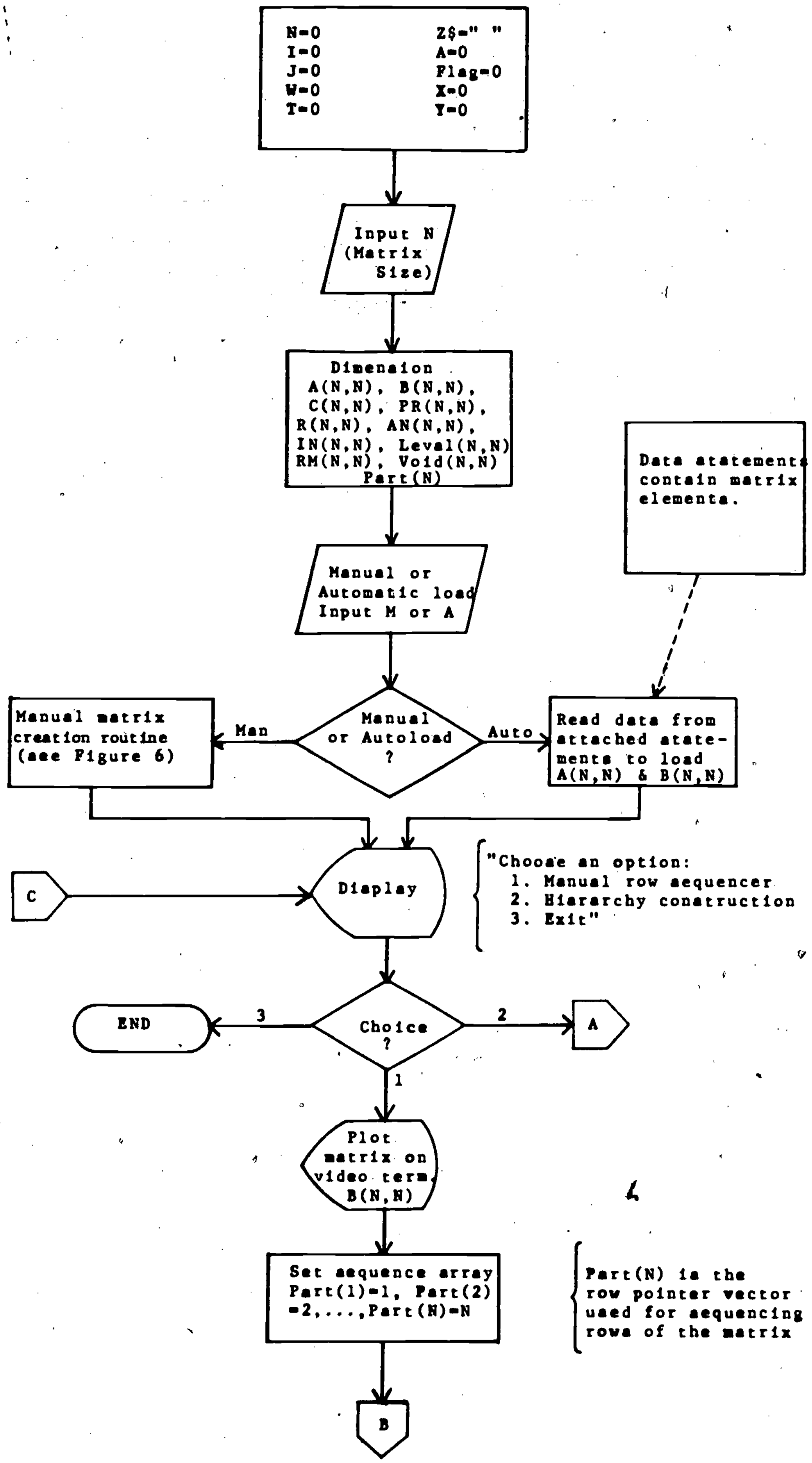
19. Tatsuoka, Maurice M. "Recent Psychometric Developments in Japan: Engineers Grapple with Educational Measurement Problems," presented at ONR Contractor's Meeting on Individualized Measurement, Columbia, Missouri, September 19, 1978, p. 6

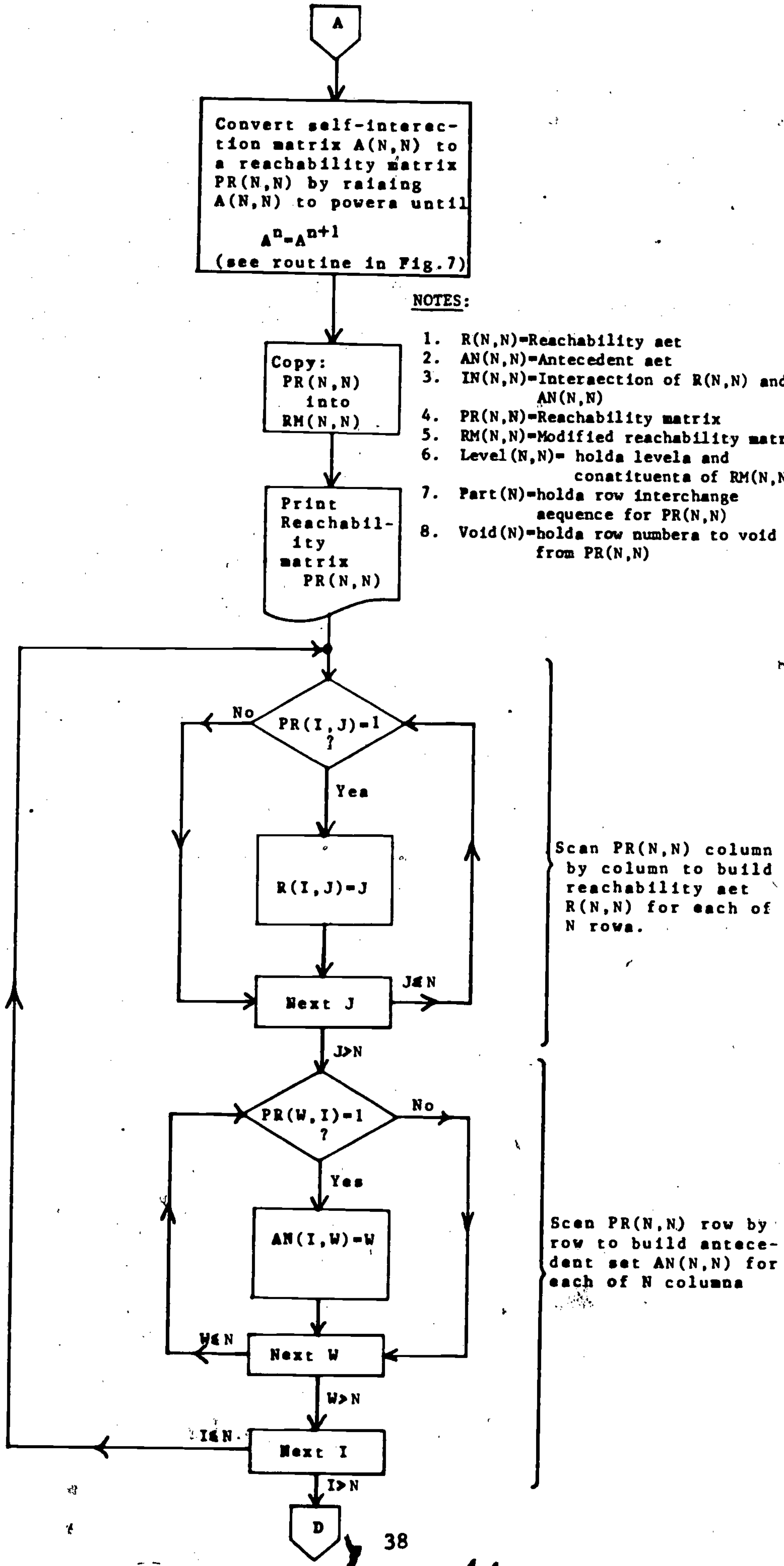
20. Op. Cit., Warfield, Structuring Complex Systems, Chapter 2, p. 9

21. Tatsuoka, Maurice M. personal communications, June 1981; (reply) July 1981

APPENDIX A

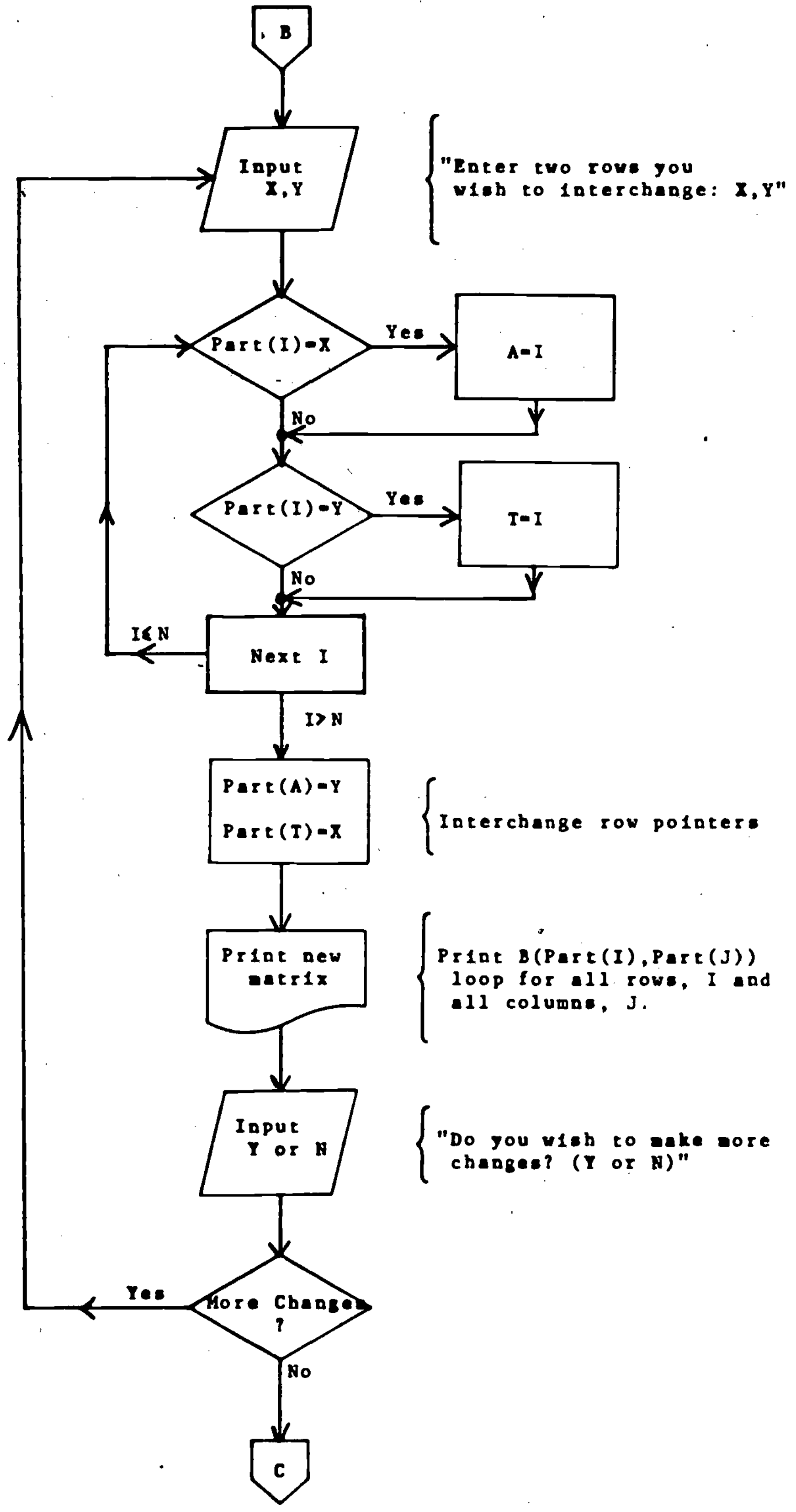
A Detailed Computer Flowchart for Developing
A Sequence Digraph From a Set of
Curriculum Objectives

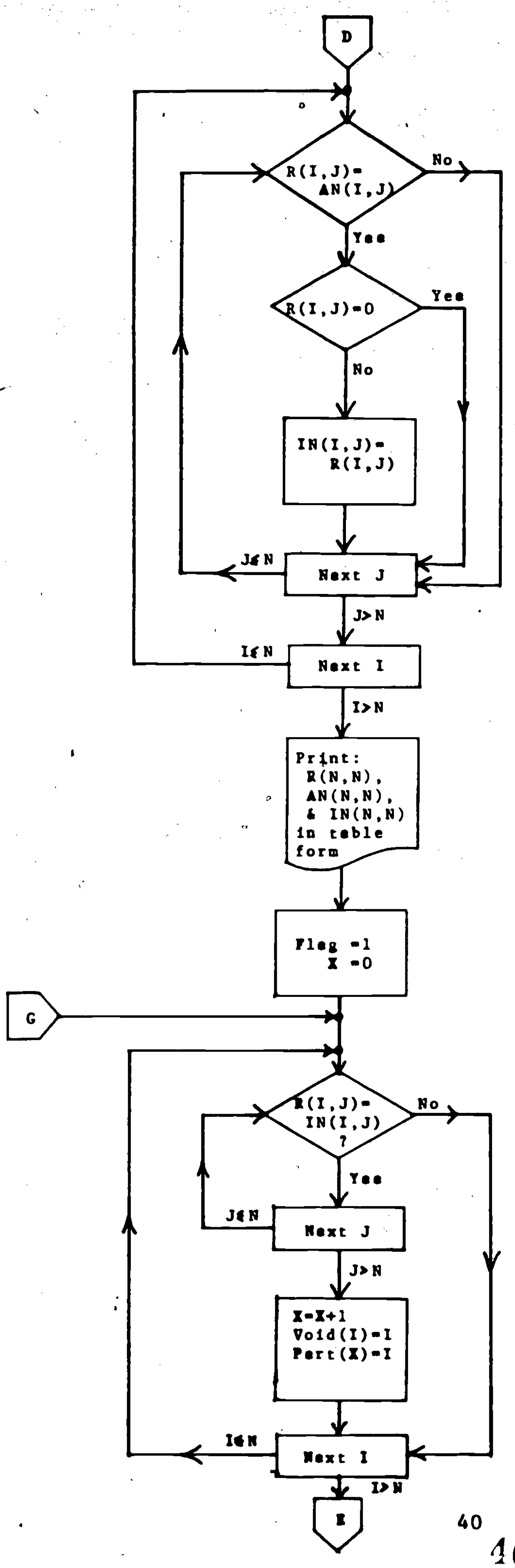

N=0
I=0
J=0
W=0
T=0
Z$=" "
A=0
Flag=0
X=0
Y=0
Input N (Matrix Size)
Dimension A(N,N), B(N,N), C(N,N), PR(N,N), R(N,N), AN(N,N), IN(N,N), Leval(N,N) RM(N,N), Void(N,N) Part(N)
Data statements contain matrix elements.
Manual or Automatic load Input M or A
Manual matrix creation routine (see Figure 6)
Man
Manual or Autoload ?
Auto
Read data from attached statements to load A(N,N) & B(N,N)
C
Display
"Choose an option:
1. Manual row sequencer
2. Hiararchy construction
3. Exit"
END
3
Choice ?
2
A
1
Plot matrix on video term. B(N,N)
Set sequence array Part(1)=1, Part(2) =2,...,Part(N)=N
Part(N) is the row pointer vector used for sequencing rows of the matrix
B


A
Convert self-interaction matrix A(N,N) to a reachability matrix PR(N,N) by raising A(N,N) to powers until
$A^n = A^{n+1}$
(see routine in Fig.7)
Copy: PR(N,N) into RM(N,N)
Print Reachability matrix PR(N,N)
PR(I,J)=1 ?
No
Yes
R(I,J)=J
Next J
J≤N
J>N
Scan PR(N,N) column by column to build reachability set R(N,N) for each of N rows.
PR(W,I)=1 ?
No
Yes
AN(I,W)=W
Next W
W≤N
W>N
Scan PR(N,N) row by row to build antecedent set AN(N,N) for each of N columns
Next I
I≤N
I>N
D

NOTES:

1. R(N,N)=Reachability set
2. AN(N,N)=Antecedent set
3. IN(N,N)=Intersection of R(N,N) and AN(N,N)
4. PR(N,N)=Reachability matrix
5. RM(N,N)=Modified reachability matrix
6. Level(N,N)= holds levels and constituents of RM(N,N)
7. Part(N)=holds row interchange sequence for PR(N,N)
8. Void(N)=holds row numbers to void from PR(N,N)


B
Input X,Y
"Enter two rows you wish to interchange: X,Y"
Part(I)=X
Yes
A=I
No
Part(I)=Y
Yes
T=I
No
I≤N
Next I
I>N
Part(A)=Y
Part(T)=X
Interchange row pointers
Print new matrix
Print B(Part(I),Part(J)) loop for all rows, I and all columns, J.
Input Y or N
"Do you wish to make more changes? (Y or N)"
Yes
More Changes ?
No
C

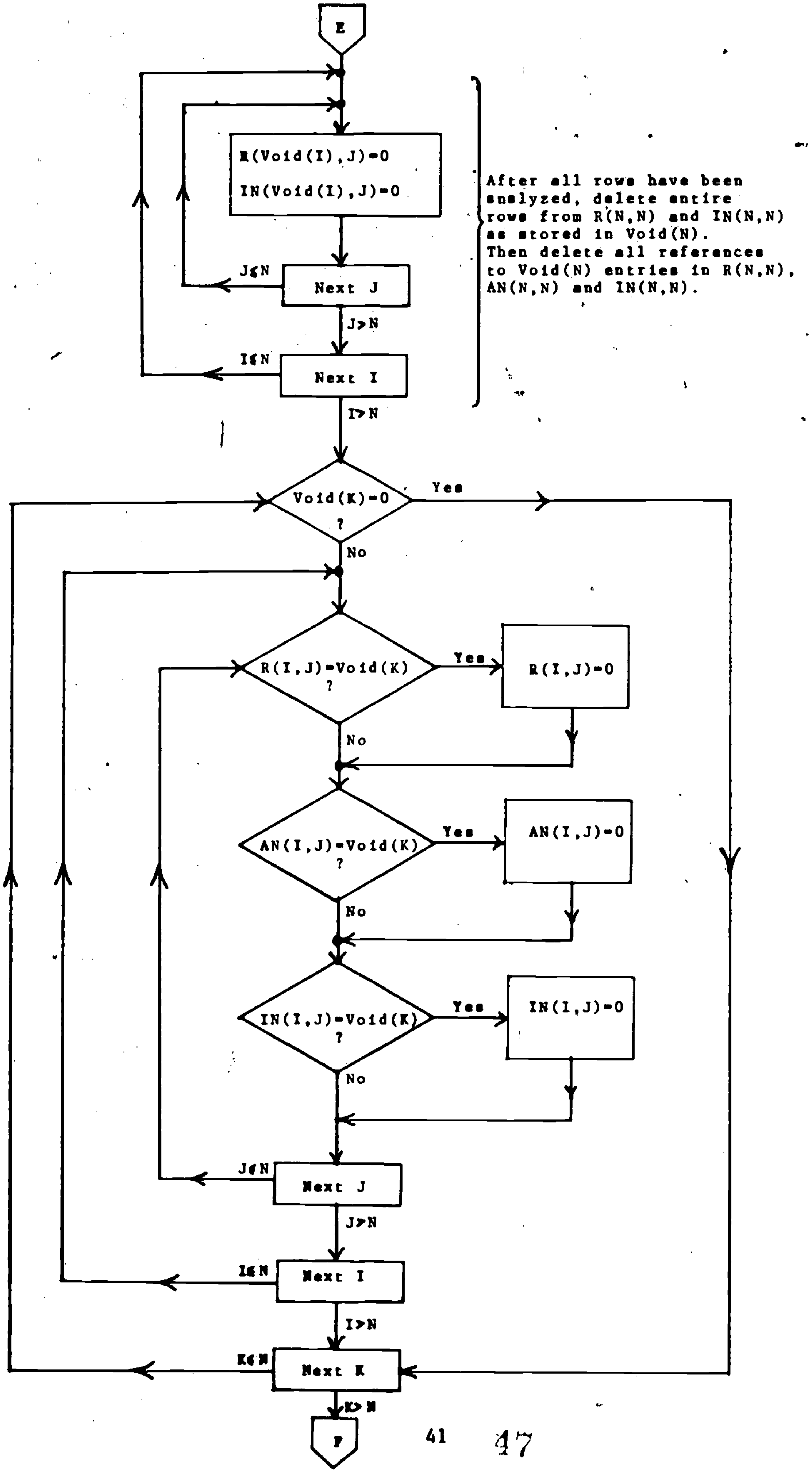


Compute intersection of R(N,N) and AN(N,N)

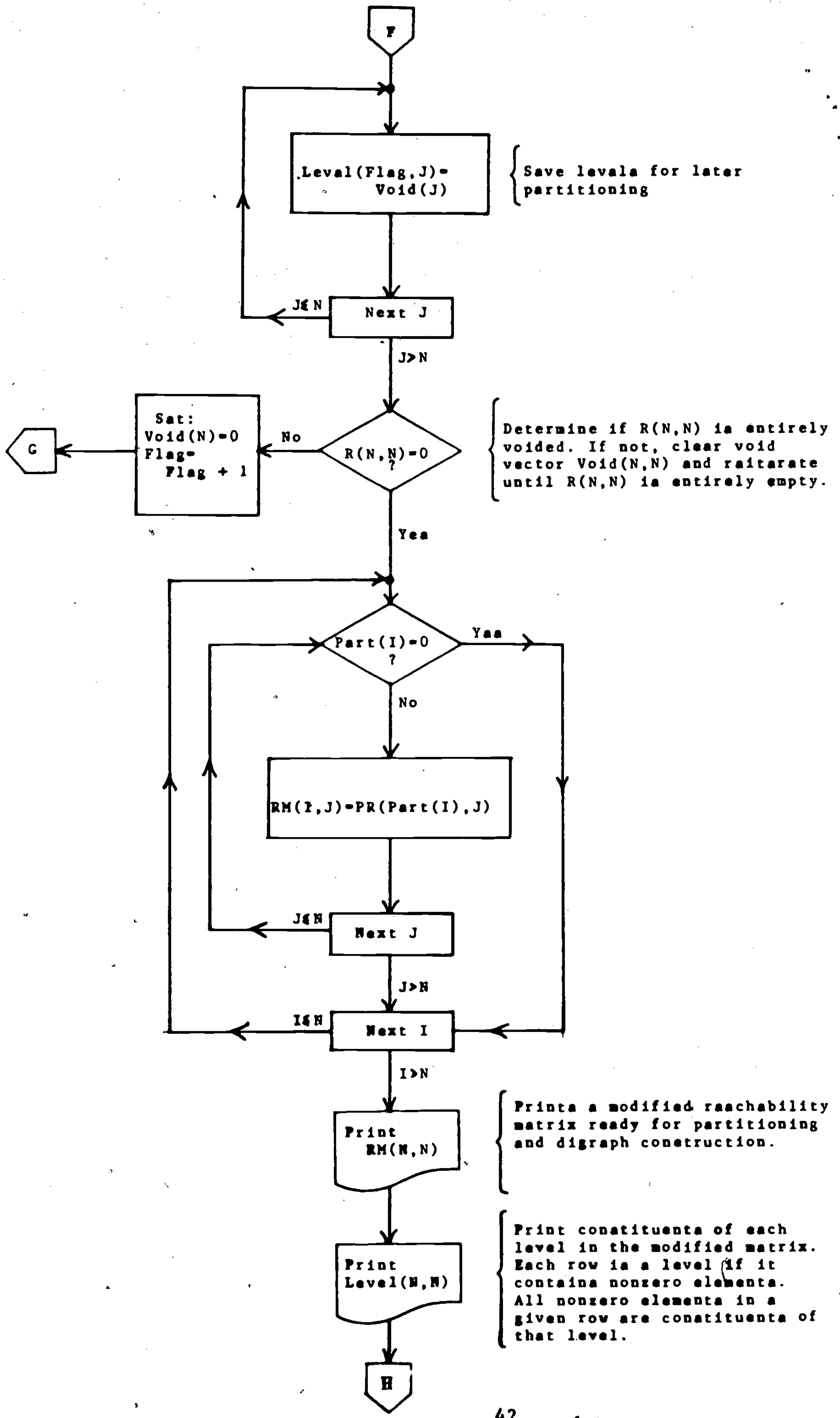
****
Reachability table computed
****

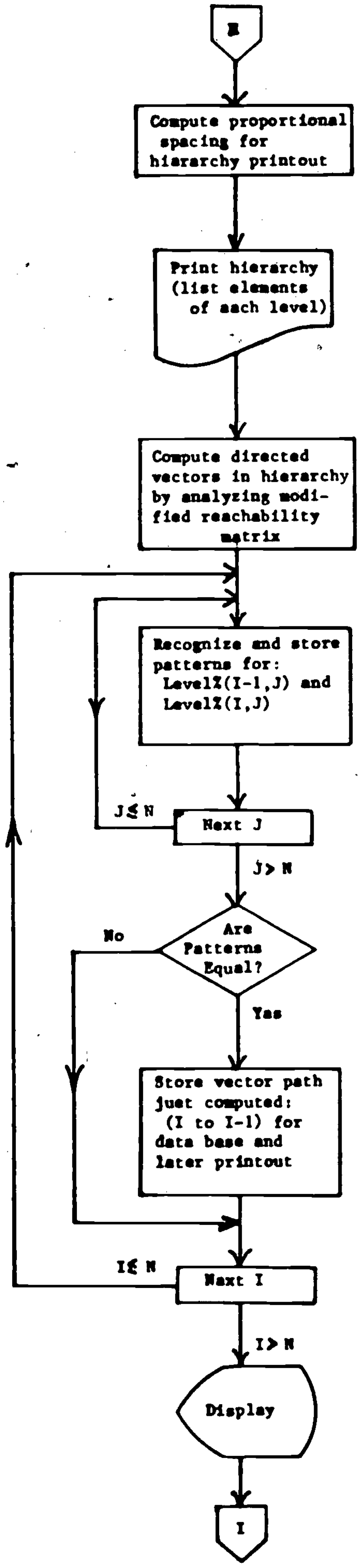
Compare R(N,N) with IN(N,N) column by column for each row. If corresponding rows are equal, store row number for voiding from reachability table.

At this point, there is a match on row I between R(I,N) and IN(I,N). store row I and reiterate for all N rows.



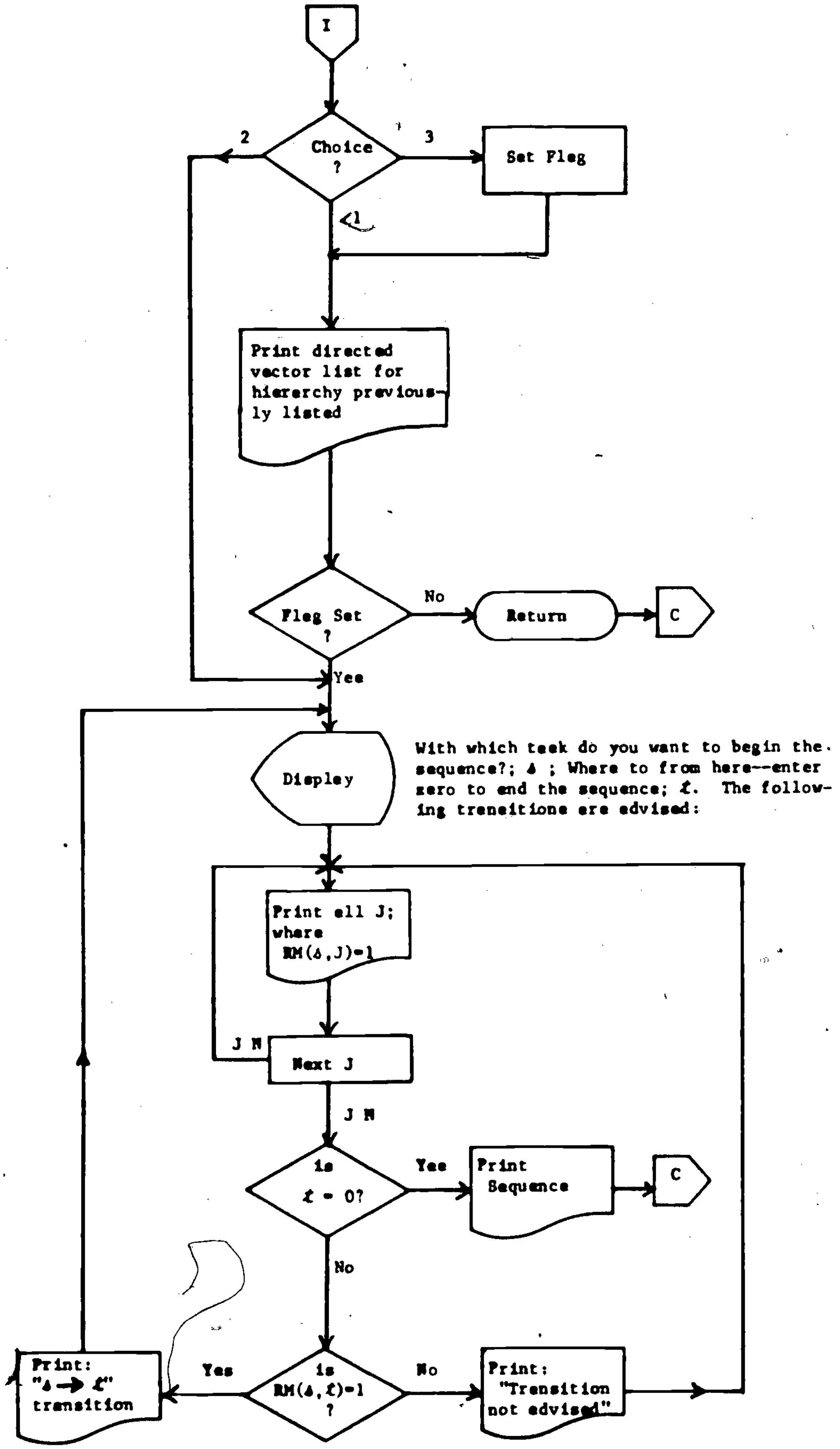
After all rows have been analyzed, delete entire rows from R(N,N) and IN(N,N) as stored in Void(N). Then delete all references to Void(N) entries in R(N,N), AN(N,N) and IN(N,N).


F
Leval(Flag,J)= Void(J)
Save levals for later partitioning
J≤N
Next J
J>N
Sat: Void(N)=0 Flag= Flag + 1
No
R(N,N)=0 ?
G
Determine if R(N,N) is entirely voided. If not, clear void vector Void(N,N) and reiterate until R(N,N) is entirely empty.
Yes
Part(I)=0 ?
Yes
No
RM(I,J)=PR(Part(I),J)
J≤N
Next J
J>N
I≤N
Next I
I>N
Print RM(N,N)
Prints a modified reachability matrix ready for partitioning and digraph construction.
Print Level(N,N)
Print constituents of each level in the modified matrix. Each row is a level if it contains nonzero elements. All nonzero elements in a given row are constituents of that level.
H


H
Compute proportional spacing for hierarchy printout
Print hierarchy (list elements of each level)
Compute directed vectors in hierarchy by analyzing modified reachability matrix
Recognize and store patterns for: Level%(I-1,J) and Level%(I,J)
J≤N
Next J
J>N
Are Patterns Equal?
No
Yes
Store vector path just computed: (I to I-1) for data base and later printout
I≤N
Next I
I>N
Display
I

NOTE: The Ith level pattern in Level% matrix is compared with the (I-1)th pattern in Level% for all elements looking for a match.

Choice:
1. List of directed vectors
2. Create a curriculum sequence
3. Both


I
Choice ?
2
3
1
Set Flag
Print directed vector list for hierarchy previously listed
Flag Set ?
No
Return
C
Yes
Display
With which task do you want to begin the sequence?; s ; Where to from here--enter zero to end the sequence; ℓ. The following transitions are advised:
Print all J; where RM(s,J)=1
J N
Next J
J N
is ℓ = 0?
Yes
Print Sequence
C
No
is RM(s,ℓ)=1 ?
Yes
No
Print: "s → ℓ" transition
Print: "Transition not advised"

APPENDIX B

An APPLESOFT BASIC © Program Listing
Based Upon the Flowchart of APPENDIX A

NOTE: The program is divided into four (4) parts and filed onto disc by their appropriate names:

1. Router Routine
2. Matrix Maker
3. Chase I
4. Chase II

Each program segment calls the next from the disc.

```
5  REM  GREETING/ROUTER ROUTINE
10  HOME
15 D$ =  CHR$ (4)
20  VTAB 8: HTAB 15
30  INVERSE : PRINT "CHASE": NORMAL
40  PRINT : PRINT "      CURRICULUM"
50  PRINT "       HIERARCHY"
60  PRINT "         AND"
70  PRINT "          SEQUENCING"
80  PRINT "           EXPERIMENT"
90  VTAB 19: PRINT "(C)  COPYRIGHT 1981 BY TOM RENCKLY
100  HTAB 26: PRINT "& GARY ORWIG"
120  FOR PAUSE = 1 TO 5000: NEXT PAUSE
125  HOME
126  VTAB 12: HTAB 15
130  PRINT : PRINT D$;"RUN MATRIX MAKER "
140  END
```

```
1000  REM  MATRIX MAKER
1010  HOME
1020  PRINT : PRINT : PRINT
1030  PRINT "SELECT ONE:"
1040  PRINT : PRINT
1050  PRINT "    1. GET AND RUN DATA SET ON DISC"
1060  PRINT
1070  PRINT "    2. START NEW DATA SET"
1080  PRINT
1090  PRINT "    3. REVIEW / CHANGE DATA SET ON DISC"
1100  PRINT
1110  PRINT "    4. QUIT"
1120  GET GA$
1130 GA =  VAL (GA$)
1135  IF GA$ = "D" GOTO 2010
1140  IF GA < 1 OR GA > 4 THEN 1120
1150  ON GA GOTO 2000,3000,4000,11000
1160  REM
1170  REM
2000  REM  GET AND RUN DATA SET FROM DISC
2010  PRINT
2020  PRINT  CHR$ (4);"RUN CHASE 1"
```

```
3000  REM  START NEW DATA SET
3010  RUN 3020
3020  HOME : PRINT "HOW MANY OBJECTIVES ARE IN THIS  UNIT";
3030  INPUT N
3040  DIM TN$(N),A%(N,N)
3050  FOR I = 1 TO N
3060  HOME
3070  PRINT "WHAT IS THE NAME OF OBJECTIVE ";I;"?"
3080  PRINT : PRINT "ENTER A 25 CHARACTER OR LESS DESCRIPTION"
3090  VTAB 8: HTAB 26: PRINT "^"
3100  PRINT "                              25 CHARS"
3110  VTAB 7
3120  INPUT SA$
3130 SA$ =  LEFT$ (SA$,25)
3140  VTAB 7
3150  PRINT "
3160  VTAB 7: PRINT " ";SA$
3170  FOR L = 1 TO 500: NEXT L
3180 TN$(I) = SA$
3190  NEXT I
3200  REM  CREATE MATRIX
3210  HOME
3220  FOR I = 1 TO N
3230  FOR J = 1 TO N
3240  IF I = J THEN 3360
3250  HOME
3260  VTAB 6
3270  PRINT "IS MASTERY OF OBJECTIVE NO.  ";I
3280  PRINT : PRINT TN$(I)
3290  PRINT : PRINT "NECESSARY FOR ACCOMPLISHMENT OF ": PRINT "OBJECTIVE N
     O.  ";J
3300  PRINT : PRINT TN$(J);"?"
3320  GET SA$
3330  IF SA$ = "Y" THEN 3360
3340  IF SA$ = "N" THEN 3380
3350  GOTO 3320
3360 A%(I,J) = 1
3370  GOTO 3390
3380 A%(I,J) = 0
3390  NEXT J
3400  NEXT I
3410  REM  DISPLAY NAMES AND MATRIX
3420  HOME
3430  FOR I = 1 TO N: PRINT I;". ";TN$(I): NEXT I
3440  PRINT : PRINT
3450  FOR B = 1 TO 2
3453  PRINT "    ";: REM  4 SPACES
3456  FOR I = 1 TO N
3459  IF B = 1 AND I < 10 THEN  PRINT " ";: GOTO 3471
3462  IF B = 1 AND I > 9 THEN  PRINT  INT (I / 10);: GOTO 3471
3465  IF B = 2 AND I < 10 THEN  PRINT I;: GOTO 3471
3468  IF B = 2 AND I > 9 THEN  PRINT (I - ( INT (I / 10) * 10));
3471  NEXT I
3474  PRINT
3477  NEXT B
3480  PRINT
3483  FOR I = 1 TO N
3486  PRINT I;".";
3489  FOR J = 1 TO N
3492  PRINT  TAB( 5);A%(I,J);
3495  NEXT J
3498  PRINT
3501  NEXT I
3520  PRINT : PRINT "PRESS RETURN TO CONTINUE"
3530  INPUT SA$
3540  HOME : VTAB 6: PRINT "SELECT ONE OF THE FOLLOWING:"
```

```
3550  PRINT : PRINT : PRINT
3560  PRINT "     1. SAVE THE MATRIX ONTO DISK"
3570  PRINT
3580  PRINT "     2. MAKE CHANGES  "
3590  PRINT
3600  PRINT "     3. DISCARD MATRIX AND RETURN"
3610  PRINT "        TO MAIN MENU
3620  PRINT
3630  GET SA$
3640  IF  VAL (SA$) < 1 OR  VAL (SA$) > 3 THEN 3630
3650  ON  VAL (SA$) GOTO 3680,3660,3860
3660  REM  SAVE BEFORE CHANGING
3670 FL = 1: REM  FLAG FOR CHANGE MATRIX
3680  PRINT : PRINT "CATALOG"
3690  IF FL = 1 THEN  PRINT : PRINT "JUST TO BE SAFE WE WILL SAVE THE MATR
    IX"
3700  PRINT : PRINT "WHAT WOULD YOU LIKE TO NAME THIS MATRIX?"
3710  INPUT UT$
3720  PRINT "OPEN ";UT$
3730  PRINT "DELETE ";UT$
3740  PRINT "OPEN ";UT$
3750  PRINT "WRITE";UT$
3760  PRINT N
3770  FOR I = 1 TO N: PRINT TN$(I): NEXT I
3780  FOR I = 1 TO N
3790  FOR J = 1 TO N
3800  PRINT A%(I;J)
3810  NEXT J
3820  NEXT I
3830  PRINT "CLOSE ";UT$
3840  PRINT : PRINT : PRINT "MATRIX ";UT$;" HAS BEEN SAVED."
3850  IF FL = 1 THEN 4500
3860  RUN : REM  RESTART TO ALLOW REDIMENSION
4000  REM  REVIEW/CHANGE DATA FROM DISK
4010  RUN 4020: REM    RESTART TO RESET DIM STATEMENTS - GET DATA FILE
4020  PRINT
4030  PRINT "CATALOG"
4040  PRINT : PRINT "WHAT IS THE NAME OF THE DATA SET?"
4050  INPUT UT$
4055  IF  LEN (UT$) = 0 THEN 4050
4060  PRINT "OPEN ";UT$
4070  PRINT "READ ";UT$
4080  INPUT N
4090  DIM TN$(N),A%(N,N)
4100  FOR I = 1 TO N: INPUT TN$(I): NEXT I
4110  FOR I = 1 TO N
4120  FOR J = 1 TO N
4130  INPUT A%(I,J)
4140  NEXT J
4150  NEXT I
4160  PRINT "CLOSE ";UT$
4170  REM  READY TO REVIEW/CHANGE
4500  REM  REVIEW / CHANGE DATA
4510 FL = 0
4520  HOME
4530  VTAB 6
4540  PRINT "SELECT ONE OF THE FOLLOWING:"
4550  PRINT : PRINT "    1. REVIEW/CHANGE A DESCRIPTOR"
4560  PRINT : PRINT "    2. REVIEW/CHANGE MATRIX"
4570  PRINT : PRINT "    3. ADD TASK(S)"
4580  PRINT : PRINT "    4. DELETE TASK(S)"
4590  PRINT : PRINT "    5. ";: INVERSE : FLASH : PRINT "SAVE";: NORMAL : PRINT
    " FILE AFTER CHANGES"
4600  PRINT : PRINT "    6. RETURN TO MAIN MENU"
4610  PRINT
4620  GET SA$
```

```
4630  IF  VAL (SA$) < 1 OR  VAL (SA$) > 6 THEN 4620
4640  ON  VAL (SA$) GOTO 5000,6000,7000,8000,9000,10000
5000  REM  REVIEW/CHANGE A DESCRIPTOR
5010  HOME : VTAB 6
5020  FOR I = 1 TO N: PRINT I;". ";TN$(I): NEXT I
5030  PRINT
5040  PRINT "MAKE A CHANGE (Y/N)?"
5050  GET SA$
5060  IF SA$ = "N" THEN 4500
5070  IF SA$ = "Y" THEN 5090
5080  GOTO 5050
5090  PRINT "WHAT IS THE NUMBER OF THE DESCRIPTOR ": PRINT "TO BE CHANGED"
     ;
5100  INPUT SA
5110  PRINT : PRINT TN$(SA)
5120  PRINT : PRINT "TO BE CHANGED TO:"
5130  PRINT : INPUT SA$
5140 SA$ =  LEFT$ (SA$,25)
5150  PRINT SA$
5160  FOR I = 1 TO 500: NEXT I
5165 TN$(SA) = SA$
5170  PRINT "DONE": FOR I = 1 TO 500: NEXT I
5180  GOTO 4500
6000  REM  REVIEW/CHANGE THE MATRIX
6002  FOR B = 1 TO 2
6003  PRINT "    ";: REM  4 SPACES
6004  FOR I = 1 TO N
6005  IF B = 1 AND I < 10 THEN  PRINT " ";: GOTO 6014
6006  IF B = 1 AND I > 9 THEN  PRINT  INT (I / 10);: GOTO 6014
6007  IF B = 2 AND I < 10 THEN  PRINT I;: GOTO 6014
6008  IF B = 2 AND I > 9 THEN  PRINT (I - ( INT (I / 10) * 10));
6014  NEXT I
6015  PRINT
6016  NEXT B
6017  PRINT
6020  FOR I = 1 TO N
6025  PRINT I;".";
6030  FOR J = 1 TO N
6040  PRINT  TAB( 5);A%(I,J);
6050  NEXT J
6060  PRINT
6070  NEXT I
6080  PRINT :: PRINT "'1' STANDS FOR YES, '0' STANDS FOR NO"
6090  PRINT "DO YOU WANT TO EXAMINE TWO DESCRIPTORS": PRINT "IN DETAIL (Y/
     N)";
6100  GET SA$
6110  IF SA$ = "N" THEN 4500
6120  IF SA$ = "Y" THEN 6140
6130  GOTO 6100
6140  PRINT : PRINT : PRINT "ENTER THE NUMBERS OF THE TWO DESCRIPTORS": PRINT
     "LIKE THIS: 5,7 AND PRESS RETURN"
6150  INPUT S1,S2
6160  HOME
6170  VTAB 6: PRINT "                1.": PRINT
6180  PRINT "IS ";TN$(S1);" REQUIRED"
6190  PRINT "BEFORE ";TN$(S2)
6200  IF A%(S1,S2) = 1 THEN 6220
6210  PRINT : PRINT "NO": GOTO 6230
6220  PRINT : PRINT "YES"
6230  PRINT : PRINT : PRINT "                2.": PRINT
6240  PRINT "IS ";TN$(S2);" REQUIRED"
6250  PRINT "BEFORE ";TN$(S1)
6260  IF A%(S2,S1) = 1 THEN 6280
6270  PRINT : PRINT "NO": GOTO 6290
6280  PRINT : PRINT "YES"
6290  PRINT : PRINT "DO YOU WANT TO CHANGE EITHER OF THESE?";
```

```
6300  GET SA$
6310  IF SA$ = "N" THEN 4500
6320  IF SA$ = "Y" THEN 6340
6330  GOTO 6300
6340  PRINT : PRINT "ENTER 1 OR 2"
6350  GET SA
6360  IF SA < 1 OR SA > 2 THEN 6350
6370  IF SA = 2 THEN 6410
6380  IF A%(S1,S2) = 0 THEN 6400
6390 A%(S1,S2) = 0: GOTO 6160
6400 A%(S1,S2) = 1: GOTO 6160
6410  IF A%(S2,S1) = 0 THEN 6430
6420 A%(S2,S1) = 0: GOTO 6160
6430 A%(S2,S1) = 1: GOTO 6160
7000  REM  ADD TASKS
7005  HOME
7010  FOR I = 1 TO N: PRINT TN$(I): NEXT I
7012  PRINT
7013  PRINT " DO YOU WANT TO ADD": PRINT "OBJECTIVES (Y/N)?
7014  GET SA$
7015  IF SA$ = "N" THEN 4500
7016  IF SA$ = "Y" THEN 7018
7017  GOTO 7014
7018  PRINT : PRINT "ADD HOW MANY TASKS";: INPUT SA
7020 N = N + SA
7025  PRINT : PRINT : PRINT "PLEASE WAIT WHILE I RESET MY DIMENSIONS"
7030  PRINT "OPEN SETUP"
7035  PRINT "DELETE SETUP"
7040  PRINT "OPEN SETUP"
7045  PRINT "WRITE SETUP"
7050  PRINT N
7055  PRINT UT$
7060  PRINT "CLOSE SETUP"
7065  RUN 7070
7070  HOME
7075  PRINT
7080  PRINT "OPEN SETUP"
7085  PRINT "READ SETUP"
7090  INPUT N
7095  DIM TN$(N),A%(N,N)
7100  INPUT UT$
7105  PRINT "CLOSE SETUP"
7110  PRINT "OPEN ";UT$
7115  PRINT "READ ";UT$
7120  INPUT N1
7125  FOR I = 1 TO N1: INPUT TN$(I): NEXT I
7130  FOR I = 1 TO N1
7135  FOR J = 1 TO N1
7140  INPUT A%(I,J)
7145  NEXT J
7150  NEXT I
7155  PRINT "CLOSE ";UT$
7160  FOR I = 1 TO N
7165  PRINT I;" .";TN$(I)
7170  NEXT I
7175  PRINT : PRINT : PRINT "LET'S WORK WITH ONE TASK AT A TIME."
7180  PRINT "YOU MAY ADD A TASK AFTER ANY OF THE"
7185  PRINT "CURRENT TITLES.  I WILL LOWER THE REST."
7190  PRINT
7210  PRINT : PRINT "AFTER WHICH CURRENTLY USED NUMBER";
7215  INPUT SA
7220  IF SA < 0 OR SA > N1 THEN GOTO 7215
7225 N1 = N1 + 1
7230  PRINT : PRINT "WHAT IS THE TITLE OF THE NEW TASK"
7235  INPUT SA$
7240 SA$ =  LEFT$ (SA$,25)
```

```
7245  FOR I = N TO SA + 2 STEP  - 1
7250 TN$(I) = TN$(I - 1)
7255  NEXT I
7260 SA = SA + 1
7265 TN$(SA) = SA$
7270  FOR I = N1 TO 1 STEP  - 1
7275  FOR J = N1 TO SA + 1 STEP  - 1
7280 A%(I,J) = A%(I,J - 1)
7285  NEXT J
7290  NEXT I
7295  FOR J = N1 TO 1 STEP  - 1
7300  FOR I = N1 TO SA + 1 STEP  - 1
7305 A%(I,J) = A%(I - 1,J)
7310  NEXT I
7315  NEXT J
7320  FOR I = 1 TO N1
7325 A%(I,SA) = 0
7330  NEXT I
7335  FOR J = 1 TO N1
7340 A%(SA,J) = 0
7345  NEXT J
7350  REM  FILL IN MATRIX
7355  REM
7355  HOME
7365  FOR I = 1 TO N1
7370  IF I = SA GOTO 7500
7375  HOME
7380  VTAB 6
7385  PRINT "IS MASTERY OF OBJECTIVE NO.  ";I
7390  PRINT : PRINT TN$(I)
7395  PRINT "NECESSARY FOR ACCOMPLISHMENT OF"
7397  PRINT "OBJECTIVE NO.  ";SA
7400  PRINT : PRINT TN$(SA);"?"
7410  GET SA$
7415  IF SA$ = "Y" THEN 7430
7420  IF SA$ = "N" THEN 7440
7425  GOTO 7410
7430 A%(I,SA) = 1
7435  GOTO 7445
7440 A%(I,SA) = 0
7445  HOME
7450  VTAB 6
7455  PRINT "IS MASTERY OF OBJECTIVE NO.  ";SA
7460  PRINT : PRINT TN$(SA)
7465  PRINT "NECESSARY FOR ACCOMPLISHMENT OF"
7467  PRINT "OBJECTIVE NO.  ";I
7470  PRINT : PRINT TN$(I);"?"
7480  GET SA$
7485  IF SA$ = "Y" THEN 7500
7490  IF SA$ = "N" THEN 7510
7495  GOTO 7480
7500 A%(SA,I) = 1
7505  GOTO 7515
7510 A%(SA,I) = 0
7515  NEXT I
7517  IF N1 = N THEN 4500
7520  HOME
7525  GOTO 7160
8000  REM  DELETE TASKS AND MATRIX SECTIONS
8010  HOME
8020  VTAB 6
8030  FOR I = 1 TO N: PRINT I;". ";TN$(I): NEXT I
8040  PRINT "LET'S WORK WITH ONE OBJECTIVE AT A TIME"
8050  PRINT "SHOULD I DELETE AN OBJECTIVE (Y/N)?";
8060  GET SA$
8080  IF SA$ = "N" THEN 4500
```

```
8090  IF SA$ = "Y" THEN 8110
8100  GOTO 8060
8110  PRINT
8120  PRINT "WHICH NUMBER";
8130  INPUT SA
8135  IF SA = N THEN 8280
8140  IF SA < 1 OR SA > N THEN 8130
8150  FOR I = SA TO N - 1
8160 TN$(I) = TN$(I + 1)
8170  NEXT I
8180  FOR I = 1 TO N
8190  FOR J = SA TO N - 1
8200 A%(I,J) = A%(I,J + 1)
8210  NEXT J
8220  NEXT I
8230  FOR J = 1 TO N
8240  FOR I = SA TO N - 1
8250 A%(I,J) = A%(I + 1,J)
8260  NEXT I
8270  NEXT J
8280 N = N - 1
8290  GOTO 8000
9000  REM  SAVE CHANGES
9010  HOME
9020  PRINT : PRINT "CATALOG"
9030  PRINT : PRINT "THE CURRENT NAME IS ";UT$;"."
9040  PRINT "DO YOU WANT TO SAVE THE CHANGES"
9050  PRINT "UNDER THIS NAME (Y/N)?";
9060  GET SA$
9070  IF SA$ = "Y" THEN 9200
9080  IF SA$ = "N" THEN 9100
9090  GOTO 9060
9100  PRINT : PRINT "WHAT IS THE NEW FILE NAME";
9110  INPUT UT$
9200  PRINT
9210  PRINT "OPEN";UT$
9220  PRINT "DELETE";UT$
9230  PRINT "OPEN";UT$
9240  PRINT "WRITE";UT$
9250  PRINT N
9260  FOR I = 1 TO N: PRINT TN$(I): NEXT I
9270  FOR I = 1 TO N
9280  FOR J = 1 TO N
9290  PRINT A%(I,J)
9300  NEXT J
9310  NEXT I
9320  PRINT "CLOSE";UT$
9330  PRINT : PRINT : PRINT "MATRIX ";UT$;" HAS BEEN SAVED."
9340  FOR I = 1 TO 750: NEXT I
10000  REM  RETURN TO MAIN MENU
10010  RUN
11000  REM  END
11010  PRINT "BYE FOR NOW!"
11020  END
```

```
10  REM  CHASE 1 MODULE
20  REM
30  REM
40 N = 0
50 I = 0
60 J = 0
70 W = 0
80 T = 0
90 Z$ = " "
100 D$ =  CHR$ (4)
110 E = 0
120 FLAG = 0
130 FULL = 0
140 COUNT = 0
150  HOME
160  PRINT "DO YOU HAVE A PRINTER ATTACHED:  ANSWER"
170  PRINT "Y OR N, THEN PRESS RETURN";
180  INPUT Z$
```

```
190  IF Z$ = "Y" THEN FLAG = 1
200  PRINT : PRINT D$;"CATALOG "
210  PRINT
220  PRINT "TYPE IN THE NAME OF THE DATA FILE, THEN"
230  PRINT "PRESS RETURN";
240  INPUT UT$
250  PRINT "THE FILE IS ";UT$
260  PRINT : PRINT D$;"OPEN ";UT$
270  PRINT : PRINT D$;"READ ";UT$
280  INPUT N
290  DIM A%(N,N),C%(N,N)
300  DIM PR%(N,N),LEVEL%(N,N)
310  DIM R%(N,N),AN%(N,N)
320  DIM IN%(N,N),RM%(N,N)
330  DIM VOID%(N),HOLD%(N),PART%(N)
340  DIM TN$(N)
350  FOR I = 1 TO N
360  INPUT TN$(I)
370  NEXT I
380  FOR I = 1 TO N
390  FOR J = 1 TO N
400  INPUT A%(I,J)
410 PR%(I,J) = A%(I,J)
420 C%(I,J) = A%(I,J)
430  NEXT J
440  NEXT I
450  PRINT : PRINT D$;"CLOSE ";UT$
460  IF FLAG = 1 THEN  PR# 1
470  HOME
480  VTAB 5: PRINT "SELECT ONE OF THE FOLLOWING": PRINT "THEN PRESS RETURN
     "
490  PRINT : PRINT
500  PRINT "1.  CREATE INSTRUCTIONAL SEQUENCE"
510  PRINT : PRINT "2.  INSTRUCTIONAL HIERARCHY AND SEQUENCE"
520  PRINT "3.  TECHNICAL RUN - FULL OPERATIONAL": PRINT "    PRINT"
530  GET OPT
540 K = 1
550  HOME
560  IF OPT = 3 GOTO 630
565  IF OPT < 1 OR OPT > 3 GOTO 530
570  VTAB 12: PRINT "IT WILL TAKE APPROXIMATELY ";( INT ((N * N - N * 3) *
     1.5)) / 60
580  PRINT "MINUTES TO COMPUTE YOUR HIERARCHY.  YOU"
590  PRINT "WILL BE NOTIFIED BY THE COMPUTER WHEN"
600  PRINT "THE HIERARCHY IS COMPLETED.  THANK YOU"
610  PRINT "FOR WAITING."
620  GOTO 760
630  PRINT : PRINT "AT THIS TIME, THE ORIGINAL MATRIX IS"
640  PRINT "BEING RAISED TO CONSECUTIVE POWERS."
650  PRINT "EACH ITERATION COMPARES THE CURRENT"
660  PRINT "POWER TO THE PREVIOUS ONE - SEEKING"
670  PRINT "A MATCH.": PRINT : PRINT
680  GOTO 760
690  REM  TRANSFER PR% TO C% AND REITERATE.
700  FOR I = 1 TO N
710  FOR J = 1 TO N
720 C%(I,J) = PR%(I,J)
730 PR%(I,J) = 0
740  NEXT J
750  NEXT I
760 T = 0
770  IF OPT <  > 3 GOTO 800
780  PRINT "ITERATION NUMBER ";K;".  THE MATRIX IS"
790  PRINT "CURRENTLY BEING RAISED TO THE ";(K + 1);" POWER.": PRINT
800  FOR W = 1 TO N
810  FOR I = 1 TO N
```

```
820  FOR J = 1 TO N
830 PR%(I,W) = A%(I,J) * C%(J,W)
840  IF PR%(I,W) >  = 1 GOTO 860
850  NEXT J
860  IF (C%(I,W) <  > PR%(I,W)) THEN T = T + 1
870  NEXT I
880  NEXT W
890 K = K + 1
900  IF OPT <  > 3 GOTO 930
910  PRINT "THE POWER ";K;" IS BEING COMPARED WITH"
920  PRINT "THE ";(K - 1);" POWER MATRIX": PRINT : PRINT
930  IF T > 0 GOTO 700
935  IF OPT <  > 3 GOTO 1060
940  REM  MATRIX PRINTOUT
950  IF FLAG = 1 THEN  PR# 1:Z$ =  CHR$ (12): PRINT Z$
960  PRINT "REACHABILITY MATRIX OF ORDER ";K
970  PRINT : PRINT
980  FOR I = 1 TO N
990  FOR J = 1 TO N
1000  PRINT PR%(I,J);" ";
1010  NEXT J
1020  PRINT
1030  NEXT I
1040  PRINT
1050  PRINT " THIS MATRIX IS NOW BEING COMPUTED INTO A DIGRAPH"
1060 S = 0
1070 C = 0
1080  PRINT
1090  FOR I = 1 TO N
1100  FOR J = 1 TO N
1110  IF PR%(I,J) = 1 THEN R%(I,J) = J
1120  NEXT J
1130  NEXT I
1140  FOR W = 1 TO N
1150  FOR I = 1 TO N
1160  IF PR%(I,W) = 1 THEN AN%(W,I) = I
1170  NEXT I
1180  NEXT W
1190  FOR I = 1 TO N
1200  FOR J = 1 TO N
1210  IF R%(I,J) = AN%(I,J) AND R%(I,J) <  > 0 THEN IN%(I,J) = R%(I,J)
1220  NEXT J
1230  NEXT I
1240 W = 0
1250  FOR I = 1 TO N
1260  FOR J = 1 TO N
1270  IF R%(I,J) <  > IN%(I,J) THEN J = N:Z$ = "ADVANCE": GOTO 1310
1280  FOR T = 1 TO N
1290  IF PART%(T) = I THEN T = N:J = N:Z$ = "ADVANCE": REM  ROW # ALREADY
     EXISTS IN PART%
1300  NEXT T
1310  NEXT J
1320  IF Z$ = "ADVANCE" GOTO 1370
1330 S = S + 1
1340 W = W + 1
1350 PART%(S) = I: REM  PARTITION SEQUENCE VECTOR
1360 VOID%(W) = I: REM  VOID VECTOR
1370 Z$ = " "
1380  NEXT I
1390  FOR I = C TO S
1400 T = 0
1410  FOR J = 1 TO N
1420  IF PR%(PART%(I),J) = 1 THEN T = T + 1
1430  NEXT J
1440 HOLD%(I) = T
1450  NEXT I
```

```
1460  FOR J = 1 TO S
1470 D = 0
1480  FOR I = S TO (C + 2) STEP  - 1
1490  IF HOLD%(I) >  = HOLD%(I - 1) GOTO 1540
1500 E = HOLD%(I):A = PART%(I)
1510 HOLD%(I) = HOLD%(I - 1):PART%(I) = PART%(I - 1)
1520 HOLD%(I - 1) = E:PART%(I - 1) = A
1530 D = 1
1540  NEXT I
1550  IF D = 0 THEN J = N
1560  NEXT J
1570  FOR K = 1 TO N
1580  IF VOID%(K) = 0 GOTO 1710
1590  FOR J = 1 TO N
1600 R%(VOID%(K),J) = 0
1610 AN%(VOID%(K),J) = 0
1620 IN%(VOID%(K),J) = 0
1630  NEXT J
1640  FOR I = 1 TO N
1650  FOR J = 1 TO N
1660  IF R%(I,J) = VOID%(K) THEN R%(I,J) = 0
1670  IF AN%(I,J) = VOID%(K) THEN AN%(I,J) = 0
1680  IF IN%(I,J) = VOID%(K) THEN IN%(I,J) = 0
1690  NEXT J
1700  NEXT I
1710  NEXT K
1720 COUNT = COUNT + 1
1730  FOR J = 1 TO (S - C)
1740 LEVEL%(COUNT,J) = PART%(C + J)
1750  NEXT J
1760  FOR T = 1 TO N
1770 VOID%(T) = 0
1780  NEXT T
1790  FOR T = 1 TO N
1800  FOR J = 1 TO N
1810  IF R%(T,J) > 0 THEN Z$ = "ADVANCE":J = N:T = N
1820  NEXT J
1830  NEXT T
1840 C = S
1850  IF Z$ = "ADVANCE" THEN Z$ = " ": GOTO 1240
1860  REM  AT THIS POINT, THE REACHABILITY MATRIX IS PARTITIONED.
1870  FOR I = 1 TO N
1880  FOR J = 1 TO N
1890 RM%(I,J) = PR%(PART%(I),PART%(J))
1900  NEXT J
1910  NEXT I
1920  PRINT
1930  IF OPT <  > 3 GOTO 2280
1940  IF FLAG = 1 THEN  PR# 1: PRINT : PRINT : PRINT
1950  PRINT "   MODIFIED REACHABILITY MATRIX": PRINT "      ";
1960  FOR I = 1 TO N
1970  IF PART%(I) > 9 THEN  PRINT ( INT (PART%(I) / 10));" ";: GOTO 1990
1980  PRINT "  ";
1990  NEXT I
2000  PRINT : PRINT "      ";
2010  FOR I = 1 TO N
2020  IF PART%(I) > 9 THEN  PRINT (PART%(I) - ( INT (PART%(I) / 10)) * 10)
    ;" ";: GOTO 2040
2030  PRINT PART%(I);" ";
2040  NEXT I
2050  PRINT : PRINT
2060  FOR I = 1 TO N
2070  PRINT PART%(I);
2080  FOR J = 1 TO N
2090  PRINT  TAB( 6);RM%(I,J);" ";
2100  NEXT J
```

```
2110  PRINT
2120  NEXT I
2130  PRINT
2140  PRINT "THESE ARE THE LEVELS AND THEIR": PRINT "CONSTITUENTS:": PRINT
2150  FOR I = 1 TO N
2160 K = 0
2170  FOR J = 1 TO N
2180  IF LEVEL%(I,J) > 0 THEN K = K + 1
2190  NEXT J
2200  IF K > 0 THEN  PRINT "LEVEL ";I;
2210  FOR J = 1 TO N
2220  IF LEVEL%(I,J) = 0 GOTO 2240
2230  PRINT  TAB( 10);LEVEL%(I,J);" ";
2240  NEXT J
2250  PRINT
2260  NEXT I
2270  REM  COMPUTE AND PRINT HIERARCHY
2280  IF OPT = 1 GOTO 2700
2285  IF OPT = 3 GOTO 2300
2290  GOSUB 7000
2300  IF FLAG = 1 THEN E = 80: PRINT Z$: GOTO 2340
2310 E = 40
2320  PRINT : PRINT  TAB( 8);"AN ALTERNATIVE HIERARCHY": PRINT : PRINT
2330  GOTO 2350
2340  PRINT : PRINT  TAB( 29);"AN ALTERNATIVE HIERARCHY": PRINT : PRINT
2350  FOR I = 1 TO N
2360 K = 0
2370  FOR J = 1 TO N
2380  IF LEVEL%(I,J) = 0 THEN C = J
2390  IF LEVEL%(I,J) > 0 THEN K = K + 1
2400  NEXT J
2410  IF K = 0 THEN I = N: GOTO 2610
2420  REM  COMPUTE PROPORTIONAL SPACING
2430 T =  INT ((E - K * 2) / (K + 1))
2440  FOR W = 1 TO N
2450  IF LEVEL%(I,W) = 0 GOTO 2500
2460  FOR J = 1 TO T
2470  PRINT " ";
2480  NEXT J
2490  PRINT LEVEL%(I,W);
2500  NEXT W
2510  REM  ADD SPACES BETWEEN LEVELS
2520  IF FLAG = 1 GOTO 2570
2530  FOR J = 1 TO 3
2540  PRINT
2550  NEXT J
2560  GOTO 2600
2570  FOR J = 1 TO 20
2580  PRINT
2590  NEXT J
2600  NEXT I
2610  IF FLAG = 1 GOTO 2640
2620  IF FLAG = 0 THEN  PRINT "PRESS ANY KEY WHEN YOU ARE READY TO": PRINT
 "CONTINUE"
2630  GET Z$
2640 Z$ =  CHR$ (12)
2650  REM   COMPUTE DIRECTED VECTORS FOR THE HIERARCHY
2660  PRINT Z$
2670  PRINT "THE PROGRAM IS NOW DETERMINING THE"
2680  PRINT "COMMUNICATION VECTORS THAT EXIST IN"
2690  PRINT "THE HIERARCHY JUST COMPUTED.": PRINT
2700  FOR I = 1 TO N
2710  FOR J = 1 TO N
2720 C%(I,J) = 0: REM    CLEAR C% FOR VECTOR STORAGE
2730  NEXT J
```

```
2740  NEXT I
2750 E = 0
2760 FULL = 0
2770 D = N
2780 COUNT = 0
2790  FOR I = D TO 1 STEP  - 1
2800  REM  MEASURE WIDTH OF COMM. SUBMATRIX
2810  FOR J = 1 TO N
2820  IF LEVEL%(I,J) > 0 THEN COUNT = COUNT + 1
2830  NEXT J
2840  IF COUNT > 0 GOTO 2860
2850  NEXT I
2860 K = 0
2870 D = I - 1
2880 FULL = FULL + COUNT
2890  IF COUNT = 1 GOTO 3030
2900  REM  EXAMINE EACH LEVEL FOR AN INTERNAL MAXIMAL CYCLE
2910 B = N - E
2920 C = B - COUNT + 1
2930  FOR W = B TO (C + 1) STEP  - 1: REM  MASTER LOOP
2940  FOR A = (B - 1) TO C STEP  - 1: REM  CYCLE THRU ROWS
2950  FOR J = B TO C STEP  - 1: REM  CYCLE THRU COLUMNS
2960  IF RM%(W,J) <  > RM%(A,J) GOTO 3010
2970  NEXT J
2980  IF W = A GOTO 3010
2990 C%(PART%(W),PART%(A)) = 1
3000 C%(PART%(A),PART%(W)) = 1
3010  NEXT A
3020  NEXT W
3030  IF I = 1 GOTO 3300
3040  REM  MEASURE LENGTH OF COMM. SUBMATRIX
3050  FOR J = 1 TO N
3060  IF LEVEL%((I - 1),J) > 0 THEN K = K + 1
3070  NEXT J
3080 B = N - E
3090 C = N - E - COUNT
3100 E = E + COUNT
3110  FOR W = B TO (C + 1) STEP  - 1: REM  # ROWS IN COMM. SUBMATRIX
3120  FOR X = 0 TO (K - 1): REM  # COLUMNS IN COMM. SUBMATRIX (LENGTH)
3130  FOR J = 0 TO (K - 1): REM  # ROWS IN (I-1)ST LEVEL
3140 T = 0
3150  REM  TEST ROW W FOR ALL 1S IN COMM. SUBMATRIX
3160  FOR S = C TO (C - K + 1) STEP  - 1
3170  IF RM%(W,S) = 1 THEN T = T + 1
3180  NEXT S
3190 S = C - J
3200 A = W - X - COUNT
3210  IF T = K AND RM%(A,S) = 1 THEN C%(PART%(W),PART%(A)) = 1:J = K - 1:Z
     $ = "ADVANCE"
3220  IF T <  > K AND RM%(W,S) <  > RM%(A,S) THEN J = K - 1:Z$ = "ADVANCE"

3230  NEXT J
3240  IF Z$ = "ADVANCE" THEN Z$ =  CHR$ (12): GOTO 3260
3250 C%(PART%(W),PART%(A)) = 1
3260  NEXT X
3270 COUNT = COUNT - 1
3280  NEXT W
3290  GOTO 2780
3300 S = 0
3310  FOR X = 1 TO 2
3320  FOR J = 1 TO N
3330  IF LEVEL%(X,J) > 0 THEN S = S + 1: REM  COUNT # ROWS IN TOP 2 LEVELS

3340  NEXT J
3350  NEXT X
3355  IF S = N GOTO 3455
```

```
3360  FOR W = 1 TO COUNT: REM  CYCLE THROUGH EACH ROW OF TOP LEVEL
3370  FOR X = (S + 1) TO N: REM  CYCLE THROUGH EACH ROW OF ALL LEVELS BELO
     W TOP 2
3380 T = 0
3390  FOR J = 1 TO COUNT: REM  COMPARE EACH COLUMN IN THE ROW
3400  IF A%(PART%(W),PART%(J)) = A%(PART%(X),PART%(J)) THEN T = T + 1
3410  NEXT J
3420  IF T = COUNT THEN C%(PART%(X),PART%(W)) = 1: REM   ANOTHER MATCH FOU
     ND
3430  IF T <  > COUNT THEN C%(PART%(X),PART%(W)) = 0
3440  NEXT X
3450  NEXT W
3455  IF OPT = 1 THEN  GOSUB 7000
3460  PRINT : PRINT D$;"OPEN FILE1"
3470  PRINT D$;"DELETE FILE1"
3480  PRINT D$;"OPEN FILE1"
3490  PRINT D$;"WRITE FILE1"
3500  PRINT N
3510  FOR I = 1 TO N
3520  FOR J = 1 TO N
3530  PRINT C%(I,J)
3540  PRINT LEVEL%(I,J)
3550  NEXT J
3560  NEXT I
3570  FOR I = 1 TO N
3580  PRINT PART%(I)
3590  PRINT TN$(I)
3600  NEXT I
3610  PRINT FLAG
3620  PRINT D$;"CLOSE FILE1"
3630  PRINT : PRINT D$;"RUN CHASE 2"
7000  REM  ANNUNCIATOR ROUTINE
7010  POKE 770,173: POKE 771,48: POKE 772,192: POKE 773,136: POKE 774,208:
      POKE 775,5: POKE 776,206: POKE 777,1: POKE 778,3: POKE 779,240: POKE
     780,9: POKE 781,202
7020  POKE 782,208: POKE 783,245: POKE 784,174: POKE 785,0: POKE 786,3: POKE
     787,76: POKE 788,2: POKE 789,3: POKE 790,96: POKE 791,0: POKE 792,0
7030  FOR A = 1 TO 10
7040  POKE 768,50
7050  FOR I = 20 TO 1 STEP  - 4
7060  POKE 769,I
7070  CALL 770
7080  NEXT I
7090  NEXT A
7100  FOR T = 1 TO 3
7110  FOR I = 200 TO 50 STEP  - 5
7120  POKE 768,I
7130  POKE 769,10
7140  CALL 770
7150  NEXT I
7160  NEXT T
7170  RETURN
55555  END
```

```
10  REM CHASE 2 MODULE
20  REM
30 D$ =  CHR$ (4)
40  PRINT : PRINT D$;"OPEN FILE1"
50  PRINT D$;"READ FILE1"
60  INPUT N
70  DIM C%(N,N),LEVEL%(N,N),RM%(N,N)
80  DIM PART%(N),TN$(N)
90  FOR I = 1 TO N
100  FOR J = 1 TO N
110  INPUT C%(I,J)
120  INPUT LEVEL%(I,J)
130  NEXT J
140  NEXT I
150  FOR I = 1 TO N
160  INPUT PART%(I)
170  INPUT TN$(I)
180  NEXT I
190  INPUT FLAG
200  PRINT : PRINT D$;"CLOSE FILE1"
210  PRINT "SELECT AN OPTION."
220  PRINT "   1.  LIST OF DIRECTED VECTORS"
230  PRINT "   2.  CREATE A CURRICULUM SEQUENCE"
240  PRINT "   3.  EACH IN TURN"
250  PRINT "         ": GET SA
255  IF SA < 1 OR SA > 3 THEN  HOME : GOTO 210
260  IF SA = 2 GOTO 500
270  IF FLAG = 0 THEN  SPEED= 90: GOTO 290
280  HOME : PR# 1: PRINT " "
290  FOR W = N TO 1 STEP  - 1
300  FOR A = 1 TO N
310 C%(W,W) = 0
320  IF C%(PART%(W),PART%(A)) = 1 THEN  PRINT "DRAW A DIRECTED VECTOR FROM
     ";PART%(W);" TO ";PART%(A);".": PRINT
330  NEXT A
340  NEXT W
350  SPEED= 255
360  IF SA = 3 THEN SA = 0: GOTO 500
370  PRINT "PRESS ANY KEY TO CONTINUE"
380  GET Z$
390  HOME
400  VTAB 8: PRINT "CHOOSE AN OPTION BELOW"
410  PRINT
420  HTAB 5: PRINT "1.  CREATE A CURRICULUM SEQUENCE"
430  HTAB 5: PRINT "2.  RESTART THE PROGRAM"
440  IF FLAG = 1 THEN  HTAB 5: PRINT "3.  PRINTOUT DIRECTED VECTORS": HTAB
     5: PRINT "4.  QUIT"
450  IF FLAG = 0 THEN  HTAB 5: PRINT "3.  QUIT"
460  GET T
470  IF T < 1 OR T > 4 GOTO 460
475  IF FLAG = 0 GOTO 485
480  ON T GOTO 500,490,280,1240: REM  OPTIONS AVAILABLE WITH PRINTER ON

485  ON T GOTO 500,490,1240: REM  OPTIONS AVAILABLE WITH NO PRINTER
490  PRINT : PRINT D$;"RUN MATRIX MAKER"
500  IF FLAG = 0 GOTO 520
510  PR# 1
520  HOME : VTAB 5
530  PRINT "IF YOU SELECT AN OBJECTIVE FOR YOUR"
540  PRINT "SEQUENCE WHICH IS NOT A LEGITIMATE"
550  PRINT "TRANSITION, YOUR CHOICE WILL BE"
560  PRINT "FLAGGED WITH A WHITE SQUARE TO"
```

```
570  PRINT "REMIND YOU."
580  PRINT : PRINT
590  PRINT "DURING THIS SEQUENCE CREATION ROUTINE,"
600  PRINT "AFTER ENTERING AN OBJECTIVE NUMBER,"
610  PRINT "PRESS RETURN.  FOR YES/NO RESPONSES,"
620  PRINT "PRESS ONLY Y OR N.  THE ROUTINE WILL"
630  PRINT "AUTOMATICALLY CONTINUE WITHOUT THE NEED"
640  PRINT "TO PRESS RETURN."
650  PRINT : PRINT : PRINT "PRESS RETURN WHEN YOU ARE READY"
660  PRINT "TO CONTINUE"
670  GET Z$
680  HOME
682  FOR I = 1 TO N
683  FOR J = 1 TO N
685  IF I = 1 AND J = 1 GOTO 690
687  GOTO 750
690  PRINT "WITH WHICH OBJECTIVE DO YOU WANT TO": PRINT "START THE SEQUENC
     E";
700  INPUT T
710  IF T < 0 OR T > N THEN  HOME : GOTO 690
720 RM%(1,1) = T
750  PRINT : PRINT "THE FOLLOWING TRANSITIONS ARE ADVISED:": PRINT
760  INVERSE
770  FOR COUNT = 1 TO N
780  IF C%(T,COUNT) = 1 THEN  PRINT COUNT;" ";
790  NEXT COUNT
800  NORMAL
810  PRINT
820  PRINT "WHERE WOULD YOU LIKE TO TRANSITION?"
830  PRINT "ENTER ZERO TO END THE SEQUENCE."
840  INPUT W
850  IF W > N GOTO 840
860  IF W = 0 THEN I = N:J = N: GOTO 990
870  IF C%(T,W) = 1 GOTO 935
880  INVERSE
890  PRINT "THIS TRANSITION IS OUT OF SEQUENCE": PRINT "AND NOT ADVISED.":
      NORMAL
900  PRINT : PRINT "DO YOU STILL WISH TO CHOOSE IT (Y/N)"
910  GET Z$
920  IF Z$ = "N" THEN Z$ = " ": GOTO 820
930 Z$ = "NOSEQ"
935  IF I = 1 AND J = 1 THEN J = J + 1
940 RM%(I,J) = W
950  IF Z$ = "NOSEQ" THEN RM%(I,J) = RM%(I,J) + 100
960  PRINT
970  GOSUB 1140
980 T = W
990  NEXT J
1000  NEXT I
1010  REM  SEQUENCE ENDED
1020  IF FLAG = 1 THEN  PR# 1:
1030  PRINT "HERE IS THE CURRICULUM SEQUENCE YOU HAVE DETERMINED"
1040  GOSUB 1140
1050  PRINT "DO YOU WANT TO CREATE ANOTHER SEQUENCE  (Y/N)? ";
1060  GET Z$
1080  FOR I = 1 TO N
1090  FOR J = 1 TO N
1100 RM%(I,J) = 0
1110  NEXT J
1120  NEXT I
1125  IF Z$ = "N" GOTO 390
1130  GOTO 680
1140  FOR K = 1 TO N
1150  FOR E = 1 TO N
1160  IF RM%(K,E) = 0 THEN E = N:K = N: GOTO 1190
1170  IF RM%(K,E) > 100 THEN  INVERSE : PRINT RM%(K,E) - 100;: NORMAL : PRINT
```

```
         "->";: GOTO 1190
1180  PRINT RM%(K,E);"->";
1190  NEXT E
1200  NEXT K
1210  PRINT : PRINT
1220 Z$ = " ";B$ = " "
1230  RETURN
1240  HOME : VTAB 6: HTAB 10
1250  PRINT "BYE FOR NOW"
1260  PRINT : PRINT "TO RUN THIS AGAIN, TYPE---"
1270  VTAB 12: HTAB 14
1280  INVERSE
1290  PRINT "            "
1300  HTAB 14: PRINT " RUN HELLO "
1310  HTAB 14: PRINT "            "
1320  PRINT : PRINT
1330  NORMAL
1340  PRINT : PRINT "THEN PRESS RETURN"
55555  END
```